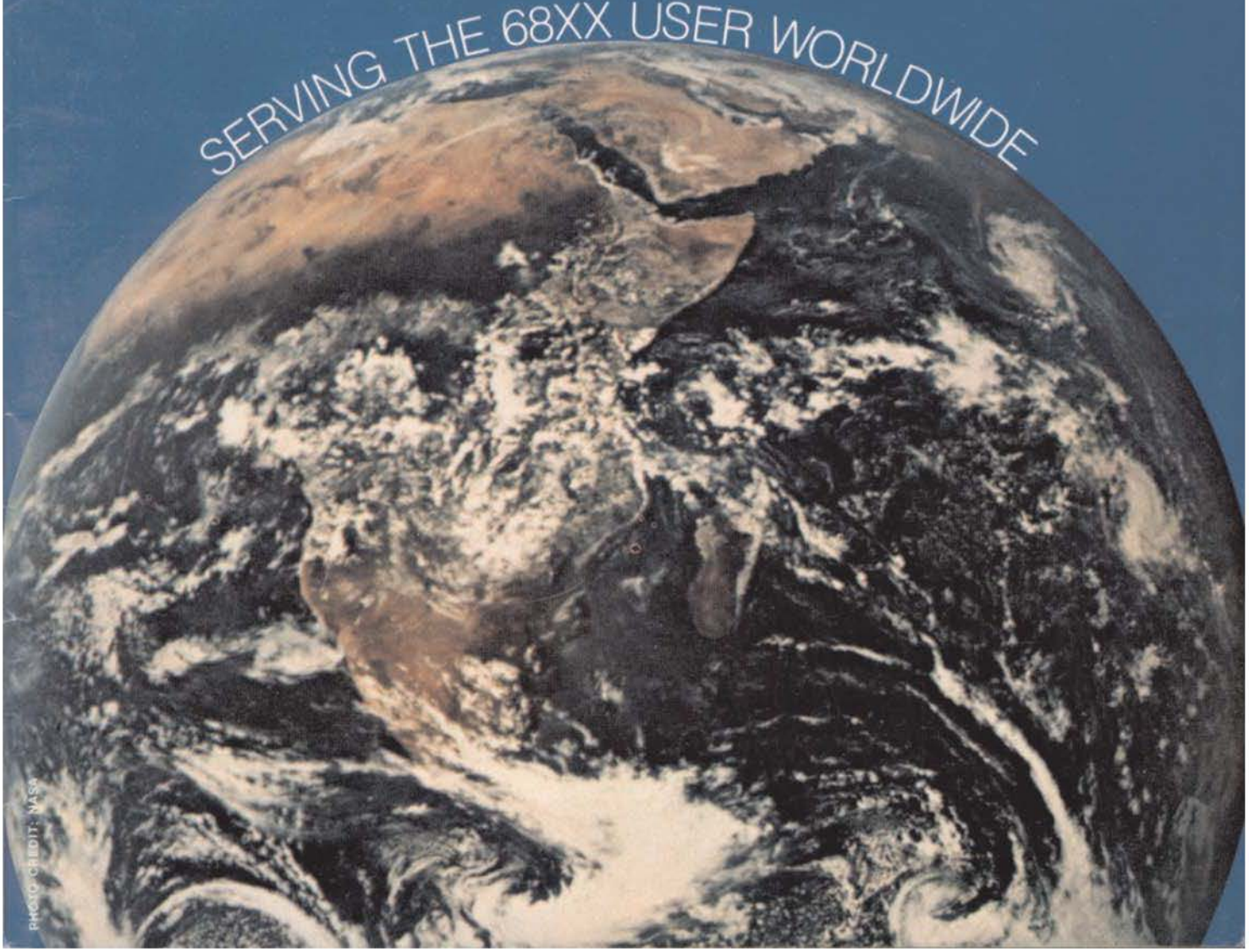
'68' MICRO JOURNAL

$2.95USA

Australia	A $ 4.75	New Zealand	NZ $ 4.75
Singapore	S $ 9.45	Hong Kong	H $23.50
Malaysia	M $ 9.45	Sweden	30:-SEK

VOLUME IV ISSUE XI • Devoted to the 68XX User • November 1982

"Small Computers Doing Big Things"



YOUR CHOICE-smart either way

- Over 140 software driven functions
- 82 x 24 or 82 x 20 screen format — software selectable
- High resolution 7 x 12 matrix characters — P-31 green phosphor
- Upper/lower case character set — plus graphics character set
- 56-key alphanumeric keyboard — plus 12-key cursor, numeric pad
- Internal editing functions — insert, delete, scroll, roll, slide, etc.
- Parallel printer I/O port
- 50 to 38,400 baud operation — programmable
- Cursor type, cursor position, print control characters, protected fields, shift inversion, dual intensity and many other features

8212 — twelve-inch diagnonal screen or **8209** — nine-inch diagnonal screen

SWTPC

SOUTHWEST TECHNICAL PRODUCTS CORPORATION
219 W. RHAPSODY
SAN ANTONIO, TEXAS 78216 (512) 344-0241

Pascal for 6809

Pascal for the 6809 is a true native code compiler. Unlike the usual P-code Pascals which run in an interpretive manner, ours produces efficient assembly language mnemonics which can be assembled and run directly. This compiler is available for both 6809 FLEX™ and UniFLEX™. Many features not found in other Pascal systems were implemented while avoiding those features completely non-standard. Features of the Pascal system include:

- Supports most of Jensen and Wirth specification
- Produces fast and efficient 6809, native code
- FLEX run-time package may be trimmed
- Double precision real numbers (16.8 digits)
- Implements scalar, subrange and structured data types
- Standard I/0 using file buffer pointers
- Dynamic storage allocation
- Ability to call other Pascal programs
- FLEX version may call assembly language programs
- Buffered or single character terminal input
- Standard math functions: SIN, COS, ARCTAN, EXP, LN, SQR, SQRT
- Random number generator function
- Many usable, sample programs included
- UniFLEX version supports:
  - Random file positioning
  - Ability to call various UniFLEX system routines
  - Ability to execute UniFLEX utility commands

Pascal on diskette for 5" and 8" 6809 FLEX is available for $200.00 The 5" version requires two disk drives. The UniFLEX version is $300.00 and includes one year of maintenance. All orders should include 3 percent for postage and handling (10 percent on foreign orders).

™FLEX and UniFLEX are trademarks of Technical Systems Consultants, Inc.

technical systems consultants, inc.

111 Providence Rd., Chapel Hill, N.C. 27514

(919) 493-1451

'68' MICRO JOURNAL

Portions of text prepared using the following.

SWTPC 6800-6809-OMAF2-COS1-CT82-Sprint 3
Southwest Technical Products
219 W. Rhapsody
San Antonio, Texas 78216

EDITOR - WORD PROCESSOR
Technical Systems Consultants, Inc.
Box 2573, W. Lafayette, IN 47906
FLEX is TM of TSC

GIMIX Super Mainframe-Assorted memory boards
GIMIX Inc.
1337 West 37th Place
Chicago, Il 60609

CONTENTS

Send All Correspondence To:

Computer Publishing Center
68 MICRO JOURNAL
5900 Cassandra Smith
PO Box 849
Hixson, TN 37343
615 842-4600

68 Micro Journal is published 12 times a year by Computer Publishing, Inc. Second class postage paid at Hixson, Tennessee and additional entries. Postmaster: send Form 3579 to 68 Micro Journal, PO Box 849, Hixson, Tennessee.

SUBSCRIPTION RATES
USA
1-Year $24.50 2-Years $42.50 3-Years $64.50
FOREIGN
See Page 52

Items Submitted for Publication

Articles submitted for publication should be accompanied by the authors full name, address, date and telephone number. It is preferred that articles be submitted on either 5 or 8 inch diskette in TSC Editor format or STYLO format. All diskettes will be returned.

The following TSC Text Processor commands ONLY should be used (due to our proportional processor): .sp space, .pp paragraph, .fi fill and .nf no fill. Also please do not format within the text with multiple spaces. The rest we will enter at time of editing.

STYLO commands are all acceptable except the .pg page command, we print edited text files in continous text.

All articles submitted on diskettes should be in TSC FLEX™ format, either FLEX2 6800, or FLEX9 6809 any version.

If articles are submitted on paper they should be on white 8X11 bond or better grade paper. No hand written articles (hand written or drawn art accepted). All paper submitted articles will be photo reproduced. This requires that they be typed or produced with a dark ribbon (no blue), single spaced and type font no smaller than 'elite' or 12 pitch. Typed text should be approximately 7 inches wide (will be reduced to column width of 3 1/2 inches). **Please use a dark ribbon!**

All letters to the editor should also comply with the above and bear a signature. Letters of 'gripes' as well as 'praise' are solicited. We attempt to publish all letters to the editor verbatim, however, we reserve the right to reject any submission for lack of 'good taste'. We reserve the right to define what constitutes 'good taste'.

Advertising: Commercial advertisers please contact the 68 Micro Journal advertising department for current rate sheet and requirements.

Classified: All classified must be non-commercial. Maximum 20 words per classified ad. Those consisting of more than 20 words should be figured at .35 cents per word. 20 words or less $7.50 minimum, one time, paid in advance. No classified ads accepted by telephone.

2MHZ 6809 SYSTEMS

GIMIX offers you a variety to choose from!

38 MB WINCHESTER SYSTEM $17,498.99

HARDWARE FEATURES:

- ★ 2MHz 6809 CPU
- ★ 512KB Static RAM
- ★ 8 RS232C Serial Ports
- ★ 2 Parallel Ports
- ★ DMA Double Density Floppy Disk Controller
- ★ Dual 8" DSDD Floppy Disk System
- ★ Dual Winchester Subsystem with Two19 MB 5¼" Winchester Drives

SOFTWARE FEATURES:

- ★ OS-9 LEVEL TWO Multi-User Operating System
- ★ OS-9 Debugger
- ★ OS-9 Text Editor
- ★ OS-9 Assembler

19 MB WINCHESTER SYSTEM $8998.09

HARDWARE FEATURES:

- ★ 128K Static Ram
- ★ 2MHz 6809 CPU
- ★ 19 MB 5¼" Winchester DMA Subsystem
- ★ 4 RS232C Serial Ports
- ★ 1 MB 5¼" Floppy Disk Drive
- ★ DMA Double Density Floppy Disk Controller

SOFTWARE FEATURES:

- ★ OS-9 LEVEL TWO Multi-User Operating System
- ★ OS-9 Text Editor
- ★ OS-9 Debugger
- ★ OS-9 Assembler

128KB MULTI-USER SYSTEM $6997.39

HARDWARE FEATURES:

- ★ 2MHz 6809 CPU
- ★ DMA Double Density Floppy Disk Controller
- ★ 128KB Static Ram
- ★ 2 RS232C Serial Ports
- ★ Dual 8" DSDD Floppy Disk System

SOFTWARE FEATURES: Your choice of either UniFLEX or OS-9 LEVEL TWO. Both are Unix-like Multi-User/Multi-Tasking Operating Systems.

56KB FLEX/OS-9 "SWITCHING" SYSTEM $4148.49

HARDWARE FEATURES:

- ★ 2MHz 6809 CPU
- ★ 56K Static Ram
- ★ 2 RS232C Serial Ports
- ★ DMA Double Density Floppy Disk Controller
- ★ 2 Built-in 5¼" 40tr DSDD Disk Drives (80 Track DSDD Drive Option . . add $400.00)

SOFTWARE FEATURES:

- ★ GMXBUG monitor — FLEX Disk Operating System
- ★ OS-9 LEVEL ONE Multi-tasking operating system for up to 56K of memory

WINCHESTER SUBSYSTEMS

Winchester packages are available for upgrading current **GIMIX** 6809 systems equipped with DMA controllers, at least one floppy disk drive, and running FLEX, OS-9 LEVEL ONE or OS-9 LEVEL TWO. The packages include one or two 19MB (unformatted) Winchester drives, DMA Hard Disk Interface, and the appropriate software drivers. The Interface can handle two 5¼" Winchester Drives, providing Automatic Data Error Detection and Correction: up to 22 bit burst error detection and 11 bit burst error correction. **UniFLEX NOW AVAILABLE**

Dual drives can be used together to provide over 30 MBytes of on line storage -- or use one for back-up of the other. (More convenient and reliable than tape backup systems.

#90 includes one 19MB Drive, Interface, and Software **$4288.90**

#91 includes two 19MB Drives, Interface, and Software. **$6688.91**

Contact GIMIX for systems customized to your needs or for more information.

50 HZ Export Versions Available

1337 WEST 37th PLACE
CHICAGO, ILLINOIS 60609
(312) 927-5510
TWX 910-221-4055

GIMIX INC.

Build performance into your system

with OS-9** software tools

Unix*-based, multitasking, modular, and versatile: these key features are some of the reasons why more 6809 computer manufacturers have selected OS-9 as their standard operating system than any other. And OS-9 has been put to work by thousands of users in almost every conceivable computer application in business, science, industry, education, and government.

Your operating system should not be a barrier between you and your computer. OS-9 is very friendly and easy to use. Its modular structure makes it easy to customize, plus its comprehensive documentation shows you exactly how to interface it to just about any I/O device.

OS-9's advanced features unleash the performance potential of almost any 6809 computer — large or small. In many respects the OS-9/6809 combination is more powerful than many *minicomputers!*

There are two basic versions of OS-9. Both have the same basic features and capabilities. OS-9 Level One runs on small to medium sized systems having up to 64K memory. The Level Two version runs on medium to large size systems having memory management hardware and up to 1 megabyte of memory, and includes record and file locking for multiuser database applications.

Here are just a few reasons why you should insist on OS-9 for your microcomputer system.

- Over 40 utility commands
- Friendly "Shell" command interpreter
- Tree-structured multilevel file directories
- Full timesharing support with log-in and file security
- Fast, secure random and sequential access files
- Comprehensive English language error messages
- Compact real-time multitasking executive
- Hardware or software memory management
- Device independent interrupt-driven I/O
- Fully ROMable for small control systems
- Standard versions available from manufacturers of most popular 6809 computers

OS-9 PASCAL Language Compiler

- most complete and versatile PASCAL available for the 6809
- capable of generating P-code for interpretive execution while debugging OR
- highly optimized 6809 assembly language source code output for maximum speed
- "virtual memory" P-code interpreter lets you run large PASCAL programs

CIS COBOL * Compiler**

- ideal for most demanding business applications
- features ISAM, Debug, ACCEPT/DISPLAY and interprogram Communications modules
- retains full compatibility with CP/M software
- meets ANSI 1974 Level One COBOL standard and is GSA certified
- Also available-FORMS 2 automatic program generator for easy interative design of screen oriented applications.

BASIC09 Structured Basic Interactive Compiler**

- fastest and most comprehensive full Basic language available for the 6809
- combines standard Basic with the best features of PASCAL
- features compiler speed, interpreter friendliness and superlative debugging facilities
- option available: Run B...a ROMable run-time system for compiled Basic 09

C Language Compiler

- complete implementation of the UNIX version 7 C language
- includes INT, CHAR, SIGNED, UNSIGNED, FLOAT AND LONG data types, structures, unions, standard C library and a full preprocessor with macro definitions
- generates fully reentrant 6809 assembly language source code output

For more information contact your computer supplier, or

MICROWARE

Microware Systems Corporation
5835 Grand Avenue, Des Moines, Iowa 50312 515-279-8844 • Telex 910-520-2535

*Unix is a trademark of Bell Laboratories. ***CIS Cobol is a trademark of Micro Focus, Inc. OS-9** and Basic09 are trademarks of Microware and Motorola, Inc.

LEVEL TWO

Expand your 6809 computer to a fast, efficient multi-user system utilizing up to one megabyte of memory, almost any I/O device, and comprehensive implementations of the most-wanted programming languages: Basic09**, C, Pascal, Cobol, and Assembler.

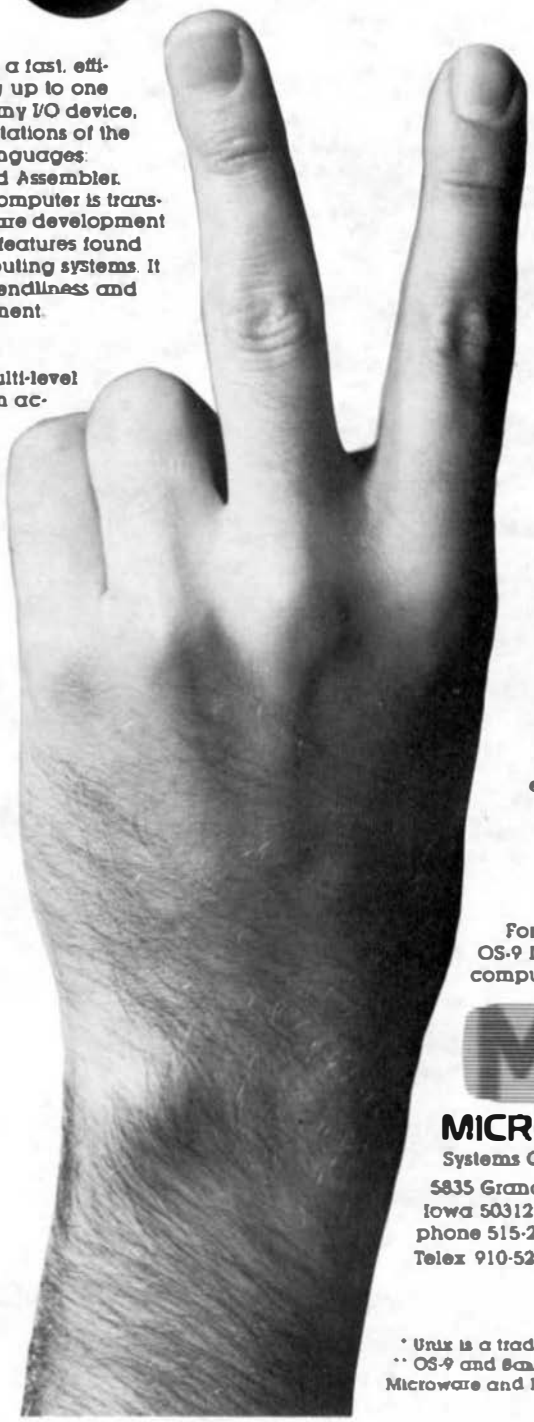
With OS-9** Level Two, your computer is transformed into the ultimate software development system with performance and features found only on large and costly computing systems. It brings to your fingertips the friendliness and power of a Unix*-style environment.

As a multi-user system...

OS-9 Level Two excels with a multi-level directory system, a fast random access file system with record lockout, user name/password logon protection, "pipes" for interprogram communication, and full file security. The versatile "Shell" command interpreter makes it easy for each user to run interactive or multiple background programs with I/O redirection to or from any file or I/O device.

As a real-time system...

Your OS-9 based computer can sense, monitor, control, and communicate with the real world thanks to OS-9's highly modular and user expandable structure. Adding customized I/O to OS-9 is almost too easy: software interfaces are simple, modular, and well documented. That is why OS-9 users routinely interface it to almost every kind of peripheral device and instrumentation.



For large systems...

OS-9 Level Two can handle over one megabyte of RAM and ROM memory with extraordinary efficiency. When, as is often the case, two or more tasks run the same program (such as Basic09) they automatically "share" just one copy in memory. Also, OS-9 Level Two typically resides in less than 24K memory. Savings like these give OS-9 based systems large capacity without having to resort to performance-robbing techniques such as disk swapping.

OS-9 Level Two is available exclusively from manufacturers of most popular 6809 computers equipped with memory management hardware. They offer versions specifically tailored to their computers for use with both new and existing systems.

For more information about OS-9 Level Two contact your computer supplier, or

MICROWARE.
Systems Corporation
5835 Grand Avenue, Des Moines,
Iowa 50312
phone 515-279-8844
Telex 910-520-2535

* Unix is a trademark of Bell Laboratories.
** OS-9 and Basic09 are trademarks of Microware and Motorola, Inc.

Turn your color computer on to the power of FLEX and OS-9

NOW FROM THE WORLDS LARGEST SUPPLIER OF SOFTWARE FOR FLEX AND OS-9 COMES FHL COLOR FLEX AND COLOR OS-9, JUST LOOK AT THESE FEATURES:

OS-9 ONLY $200
OS-9 WITH BASIC09 $299

- Unix*-based, multitasking, modular, and versatile
- Over 40 utility commands
- Friendly "Shell" command interpretor
- Tree structured multilevel file directories
- Full time sharing support with log in and file security
- Fast, secure, random and sequential access files
- Comprehensive English language error messages
- Compact real-time multitasking executive
- BASIC09 is the fastest and most comprehensive full Basic language available for the 6809. It combines standard Basic with the best features of Pascal. It's a unique interactive compiler that combines compiler speed, interpreter friendliness, and superlative debugging facilities.

FLEX NOW ONLY $99

- HI-RES screen formats
- 16 x 32 and 24 x 51, upper and lower case characters
- 24 x 64 and 32 x 64 upper case
- Full ASCII keyboards
- Easy start-up—just type RUN "FLEX"
- Online assistance just type HELP
- Optionally use a standard terminal and printer
- Advance disk I/O and terminal capabilities
- NO additional hardware required
- We have supported FLEX with more software than anyone else in the world for more than 2 years!

SPECIAL

1. DBASIC, RS Disk Basic under FLEX with a utility to copy RS to FLEX disk $30.
2. ED/ASM line and screen editor and macro assembler, both more powerful than TSC's, and at the same cost, only $100.
3. UTILITIES, a set of 12 utilities especially designed for FHL Color FLEX $50.

Get on our mailing list, call or send for our complete catalog of over 100 products for FLEX and OS-9. We're doing exciting things with your color computer!

FRANK HOGG LABORATORY

THE REGENCY TOWER • 770 JAMES ST. • SYRACUSE, NY 13203 TELEPHONE (315) 474-7856

*Unix is a trademark of Bell Labs. *OS-9 & BASIC09 are trademarks of Microware & Motorola Inc. *FLEX is a trademark of Technical Systems Consultants Inc.

DynaStar
WORD PROCESSING SYSTEM FOR OS-9

OS-9 USERS:

If your computer has a SCREEN and you're still struggling with an editor that only knows about LINES, [illegible] obviously YOU don't know about

DynaStar

DynaStar is a powerful, [illegible] equally suited to the tasks of pro[illegible] document processing. With the add[illegible] DynaForm print formatter, it is the be[illegible] package you can buy for your OS[illegible]

DynaStar Version II is now availa[illegible] nons nse "what you see is what [illegible] tually any terminal with or with[illegible] must be at least able to go to[illegible] place the cursor where you want it[illegible] able character you type is entered dir[illegible] and any non-printable control ch[illegible] mediate execution of an editing c[illegible] stroke commands permit movement [illegible] direction, by character, tab, word, line, p[illegible] and deletion of characters, words (left [illegible] whole line. Two keystroke commands aug[illegible] thi[illegible] by moving the cursor to the left margin, top[illegible] bottom[illegible] the screen, beginning or end of the edit buffer, or [illegible] beginning of the next paragraph. You can search for any string, replace with any other, do it again, mark original blocks of text, copy, move or delete blocks, read or write to side-files, set tabs and margins, or center the current line.

NOW AVAILABLE FOR FLEX

DynaStar features automatic w rd wrap, and it can right-justify text as you enter it so you will see exactly how it will look *before* you print it. If you later make alterations or change the margins, you can reform the text a paragraph at a time with two keystrokes. For pro[illegible] there is a special automatic indent mode to [illegible] te well-structured code. DynaStar includes a [illegible] lets you do almost anything (in[illegible]le) without even losing your place [illegible]ent, and it permits editing of large [illegible]hout forcing you to break up your

[illegible] powerful commands. Dyna[illegible] lets you convert any [illegible] of characters of your [illegible]o create global search-[illegible] "boiler-plate," or simply [illegible] also provide a special [illegible]rmatically executed [illegible] up modes such as [illegible] define you favorite [illegible]d.

[illegible], we offer our Dyna[illegible]hich provides all the standard [illegible]ation, headers and footers with [illegible]space, double space, multiple [illegible]face, **double-strike**, and underline. [illegible]Form[illegible] its own macro facility with string [illegible]iables, [illegible]sted include files, a full merge-print capability for generating form letters and mailing lists, and it can generate an index automatically, sorted alphabetically or by page number. You can call it from DynaStar to proof-print the active edit buffer, or by itself to print a disk file while you edit another.

Dyna[illegible]tar II: $149.95
D[illegible]naForm text formatter: $149.95
Both purchased together: $275.00
Note: DynaStar Version I (no macros) will be available at the original price until May 31, and current owners may upgrade to Version II with full credit until June 30.

AVAILABLE SOON FOR FLEX 9

Spelltest
From Dale Puckett

FOR OS-9 OR FLEX

SPELLTEST is the most versatile 68XX spelling checker available.
MENU'S MAK[illegible]OPE[illegible]IC[illegible]S[illegible] From the menu you may: Print a [illegible] list of valid words, [illegible] he; Read your text, st[illegible] ditional dictionar[illegible] ecial ap-plicati[illegible] newly ac-cepte[illegible] While c[illegible] rd. Ac-cept an[illegible] n correct spelli[illegible] Desig[illegible] right at home i[illegible] will recover the cost [illegible] 22,000 w[illegible] ntries in the America[illegible] nost common English word[illegible] 500 built in [illegible] r, [illegible] etc.) and 300 specific to your fiel[illegible] file[illegible] t and allows a large file to processed even in small computers.

NOW With A Compressed Dictionary

PRICE $199.00

A/BASIC
Basic Compiler
For OS-9 or FLEX

If you are still programming in assembler, this is the program for you! This BASIC compiler generates pure, fast efficient 6809 machine code from easy to write BASIC source programs. Uses ultra-fast integer math, extended string functions, boolean operators and run-time operations. Output is ROMable and RUNS WITHOUT ANY RUN-TIME PACKAGE. Supports IF-THEN-ELSE structure, random access, and several improvements over the original 6800 version sold by Microware. Optimized for the 6809, A/BASIC is 8 to 10 times faster than the original 6800 version, and produces code approximately 30% smaller.

SPECIAL

CHE[illegible] program coded in A/BA[illegible]IC (originally sold for $50) is included FREE on the disk in both source and object for your enjoyment. Also some utilit[illegible]s are included for testing and examples, all in [illegible]ource on the disk!

ONLY $150.00
specify OS-9 or FLEX

EDIT DISK
FOR OS-9
Examine and Modify OS-9 DISKS
Screen Orientated
Introductory Price $79.95

TRS-80 COLOR COMPUTER

FORTH FOR THE TRS-80 COLOR COMPUTER DISK SYSTEM

Trying to get [illegible] Computer?? Tired of translating HEX [illegible] of remembering where the [illegible] to program them?? Want [illegible]e code with assembly la[illegible] POKES?? Want to wri[illegible]ant to write lots of small [illegible]ut together in seconds to [illegible]e that is at least 5 to 10 [illegible] Want to learn everything the[illegible] with the best manual on th[illegible] e example of FORTH applicatio[illegible]anations of how everything works?

See Review In Nov. 80 Micro

CC FORTH IS THE ANSWER!!
Includes Editor, 6809 Assembler String Functions, Disk Data File Operations and Much Much More!

$99.95

Dynasoft RATFOR 1.4 for OS-9

Dynasoft Pascal[illegible] **Version 1.5 Now Available 40% Faster** features of the enhancements: Chain, Fread, F[illegible] Close, Delete, Fork, Send, Wa[illegible]e, Getstatus, Setstatus, SetP[illegible] This is an excellent and fast p[illegible] write utilities but powerful end[illegible]aStar.
Integer Only
Object only $69.95
Add for run-time source on disk $30.00
Add for source of Dynasoft Pascal itself $125.00

CRASMB
MULTI CPU CROSS ASSEMBLER FOR 6809 FLEX OR OS-9 by Frank Hoffman

CRASMB is a conditional macro asse[illegible]bler with the capability to use different CPU overlays in order to cross assemble. These CPU overlays called *CPU PERSONALITY MODULES (CPM's)* can be called from a source file, thereby making it easy to create object code for a variety of CPU's. It is also possible to create new *CPM's* yourself for any 8 or 16 bit CPU. The infor[illegible]ation needed is included in t[illegible]e manual. If you decide to do this, it would be advisable to purchase the source for one of the *CPM's* and modify it rather than starting from scratch. *CPM's* are currently available for the following CPU's: 6809, 6800, 6805, 6502, Z80, 8080, 1802, and others coming.

FLEX 139.95 with any CPM	OS-9 200.00 with 6809 CPM
CPM's 25.00 each	35.00 each
CPM source 25.00 each	35.00 each

Specify FLEX or OS-9 when ord[illegible]ring

OSM

For FLEX or OS-9. Create FLEX or OS-9 binary files from either FLEX or OS-9. OSM is a MACR[illegible] assembler like CRASMB. It is compatible with ISC's Assembler, but it has more powerful MACROS. OSM makes it easy to move FLEX programs to OS-9. In OS-9 it gives MACRO capability like TSC's assembler and is compatible with TSC source files. OSM was used by the author to move CRASMB to OS-9.

PRICE $125.00
Specify OS-9 or FLEX

FRANK HOGG LABORATORY
770 JAMES STREET · SYRACUSE NY 13203 Ph (315) 474-7856

Flex User Notes

Ronald W. Anderson
3540 Sturbridge Court
Ann Arbor, MI 48105

MISCELLANEOUS THOUGHTS

Help! Once more, I'm becoming overwhelmed with software. In the past week, I've received ABASIC from Frank Hogg for a test of the "tentatively final" version, Version 3 of Lucidata Pascal in its final form, (I've been involved in testing it for some time now), a copy of the latest version of Spell 'N Fix and a copy of the new Write 'N Spell from Star-Kits, and a copy of MAC95, a "macro preprocessor" from MAC Software. Recently, I received a copy of FORTH for the Color Computer, from Frank Hogg Laboratories. No complaints, mind you, just a note to the senders of such items that it might take me a while to sort through all this, and respond with comments or a brief review in the column. As I write this, it is mid Summer, and there are lots of other things to do than sit in front of the terminal every evening and weekend!

Some of you might be envious of my receiving all that good software to look at. I agree that it is usually very nice. Some of what I look at, however, is not really very good (none of those mentioned above are in the not very good category). Perhaps some of you wonder why all the "reviews" in '68' have good things to say about the software. I think Don Williams stated it some time ago, but it bears repeating. If a reviewer submits a very bad review, the first suspicion is that the software is not compatible with the particular reviewer, and it is sent to someone else. After all, we all have our individual tastes. If, however, two or more reasonably experienced software users give a particular thing a very bad review, the review is not published, and the supplier is given a chance to clean up his software. If he responds, the review is redone and published. If not, the supplier's ad is removed from '68' (if it ever appeared).

This procedure is problematical. My column is done approximately three issues ahead. I like to keep that far ahead in case of an emergency. If for some reason, I am late once in a while, there will still be a column in the mill, and no major problems will occur. advertising too, must be submitted two or three issues in advance. If you were about to enter the software (or hardware) market with a product, would you like to wait until it was absolutely finished, field tested, and ready to go, and then place your ad copy and wait three months for the ad to be run? Of course not, and neither would those suppliers, so they submit the ad copy before the product is ready for market, and hope that they can finish it and be ready for shipment when the ad is published.

Some suppliers do better than others in estimating completion time. Speaking from personal experience, software ALWAYS takes longer to complete than the most pessimistic forecast. (I have been involved in the shipping of several machines two months late, the only hold up being the software.) Now, suppose the ad runs for a couple of months, and finally the software is completed. Don's policy is not to advertise software that has not been reviewed by his staff (those of us who write regularly for '68').

Now we get the software to review and find out that it is either simply not finished, doesn't work as advertized, or for some other reason we don't consider it good enough. What would you do under those circumstances? We contact the software supplier and tell him about what we have found. He is given time to clean up his software and resubmit it for further testing. After all, we don't want to discourage anyone who is willing and able to do a good job with software but happened to skimp on the testing in the final rush of getting the thing done. Sometimes a software writer is too close to the project. He knows what dumb things not to do, but a customer who has just received the software is sure to do just those dumb things if they are not documented in the manual or made hard to do by the strategic placement of some "ARE YOU SURE?" prompts.

Few people who have never written major software have any concept of how much more rapidly the programming difficulty and time grow, than the size of the program. My estimate is that a program is a two dimensional object. (Think of a square). If you make it ten times longer, it becomes ten times wider. The total job then becomes 100 times larger. There is some justification for such a model of a program. If a program is ten times longer, it is likely to involve perhaps ten times as many variables. Now, ten times the variables related in a program ten times as long, will yield a problem 100 times as complex! A carefully segmented program, one using lots of subroutines (Procedures), need not follow this size squared complexity rule. That, in my personal opinion is the beauty of languages like Pascal, in which the problem may be broken into small parts, and each part treated like a separate small program.

The point is, that occasionally someone who has had fun for the past three years writing 300 byte Utility programs, decides to write a compiler. Generating the initial code immediately and without much planning, and little testing, gives such a person a false sense of how easy the job is going to be. He therefore gets the ad ready for publication and promises a copy of the software for review very soon.

The lack of planning yields code that is hard to follow, even for the original writer. Now, the product is nearly done, and serious testing starts. Sometimes the writer's false confidence causes him to do minimal and trivial testing and decide that the product is ready. I received such a compiler about four months ago. Virtually nothing I tried worked. I sent the supplier literally dozens of half page programs that showed up bugs in the code. At first, I sent the fixes (at least to the runtime package for which I had a source listing). I soon noted that all of his updates contained my fixes verbatim. I decided not to give away free consulting and started sending only the programs that demonstrated the bug.

The updates stopped coming. The ad was withdrawn from '68' and the supplier feels very unfairly treated by '68', Don Williams, myself, and another staff member. Unfortunately, the ad had run for three months before the software arrived for test, and a couple months after testing (because the magazine is printed several weeks ahead of distribution).

I have told the story not to scare readers away from buying software, but to indicate that '68' is very conscious of its responsibility to you readers. Also, I am not implying that lack of a review indicates a bad product. It may be simply due to lack of time to test the product (reason for my HELP! at the beginning of this column).

The compiler mentioned here is a real exception to the normal offerings of most suppliers. If you have bought this product, I strongly suspect you will have recognized it by what I have said. Send me a letter and include a page of the runtime assembler output (ie. assemble part of the runtime package and list it to your printer or better yet, include the runtime source on the disk). With that as proof, if you will also include a disk (either 5 or 8 inch, specify format single or double, 35 40 tracks etc.) I will send you my best effort at patching all the bugs in the runtime package. (The whole source file, that is.) I'd be interested in any bugs you have found. Remember that some of these are in the compiler itself. The source not being available, it would be a titan undertaking to disassemble it and fix it. Without those fixes, the compiler is usable, though not as advertized. The main problem in the compiler is in the handling of

passing parameters other than integer to functions. Without the compiler source that would be very hard to patch.

Stylograph Changes Hands

Stylograph has been bought by Control C software, related somehow to Great Plains Computer company in Idaho Falls, Idaho. They sent me a copy of the latest version some weeks ago, and I have been using it ever since. Stylograph, as you know from my previous reports, is very nice as word processing programs go, for a secretary to use in formatting reports and writing letters. The commands are very easy to learn. I learned Stylograph in about 4 hours, and am using it to write this column. There have been a few persistent bugs in Stylograph that have made many purchasers a bit unhappy. To be perfectly honest, I haven't checked this latest version to see if all of the real bugs have been cured, but it has worked very well for me.

Since I first received a copy of Stylograph from Bob Bundy, its author, to review, (he sent me a test copy of the latest version with a request to test it and report bugs, which I did), I have added an Epson MX-80 FT to my system. It has capabilities beyond the normal "TTY" printer. When I used the configuration program "STYFIX" on the new version, I set it up for the Epson. To my amazement, I can use the features that allow printing BOLDFACE, and the underline (My Epson has the GRAFTRAX enhancement).

Since I don't normally use very long files, I haven't had a chance to check out the handling of a file longer than the edit buffer, though I assume that Control C has fixed the bug that hangs up Stylograph when the last page is filled. I have a few reservations about the details of how Stylograph works, but recently have been exposed to Wordstar, the best of the text editor/formatter software for CP/M compatible systems.

I'm not impressed with Wordstar at all. It doesn't begin to compare with Stylo. There are so many control characters required, that Wordstar's author has gone to using multiple characters. The basic cursor moves use a cluster of keys much like Stylo, but the number of keystrokes required to do anything significant, is vastly larger than with Stylo. Half or more of the screen is taken up with a list of commands, which admittedly may be removed (by typing the proper control characters) so that very few lines of text file are visible at any one time. In addition, after making changes in a justified paragraph (one with even left and right margins) it is necessary to move the cursor to the beginning, and reformat it. That is necessary for each paragraph in the file. I was even more chargined to find out that in addition to all these control characters, supposedly to eliminate all the embedded commands, Wordstar has a bunch of embedded commands too! Oh well.

Stylo, of course allows you to embed the format parameters in the text, and such parameters as left margin and line length remain in effect until they are changed by embedding a different line length or margin command in the text elsewhere. Somehow that seems much more practical to me than the use of so many control codes to format the text. I'm told that I will "love Wordstar" in a few weeks or months after I get used to it, but I doubt it.

I guess what I am saying is that Stylograph, with its idiosyncrasys (in my personal opinion), is vastly better than the best of the CP/M editor formatters!

What don't I like about Stylograph? Mostly just a few quirks I have about how it could be made more usable with a few small changes. First of all, it has no OVERLAY mode. That means that you can't simply position the cursor over a letter and change it by striking a different letter. My very strong complaints to Bob Bundy early in the game, (I don't think I was the only complainer) resulted in his adding a single character overlay mode, entered by typing a "↑", which allows overtyping of the character at the cursor. There was also added provision for inserting one character by typing ↑ followed by the character to be inserted before the cursor position. Though these additions make Stylo much more useful (handling most simple typographic errors), a full overlay mode would help more. Of course this would necessitate changing all the commands to control characters, a major change in the philosophy of Stylograph.

Secondly, it works backwards (again in my opinion) with regard to deleting a section of text. In order to delete several lines, you must find the END of the section to be deleted and mark it by inserting }>. Then you must find the beginning of the text and position the cursor there. Next you use the command Z for "zap", to delete everything from the cursor to the ↑↑ mark. It seems to me that you always find the beginning of a section to be zapped first. Why not mark the beginning with {{ and put the cursor at the end? That would simplify the operation by a whole step. Same remarks hold for marking a section of text to be moved or saved to an external file.

These are perhaps personal quirks of mine, and many others may not care about these few items. These are minor complaints, and I admit to using Stylograph a great deal for text. The folks at Control C are top notch programmers, and I highly recommend Stylograph for letter writing and text processing use. It is easy to learn, and the results are very impressive, particularly with any "specialty" printer that uses a "daisy wheel" or "thimble" and proportional spacing.

It is very easy to configure the supplied Stylograph file for any terminal and printer. Most of the popular terminals are supported by simply specifying a "terminal number". Same goes for the printers. If you happen to have an oddball terminal, you may write your own file to configure Stylo to your terminal. I went through the exercise for a Televideo model 950 recently, though a Televideo configuration file is supplied. The process was easy, and the results very satisfactory. Overall, I rate Stylograph very highly.

Lucidata Pascal

For some time, I have been using a preliminary version of Lucidata Pascal version 3 for the 6809. I have come to respect Lucidata for having a solid product, and continuing to improve it. Lucidata provided the first compiler for the 68XX that I considered out of the "toy category" two or three years ago. My book, "From BASIC to Pascal" relies heavily on their compiler version 2 for the programs included in it. Now version 3 has been released. I have been using a preliminary copy for several months. Version 3 allows specification of a filename while running a program, a feature lacking in Standard Pascal. That means that you may open and close files by name within a program, and this adds greatly to the flexibility of Pascal.

Lucidata has a few other extensions to Pascal, one of which is the declaration of variables at absolute addresses, which allows access to serial and parallel ports and such things as analog to digital converters. Another extension is the inclusion of "External" functions and procedures in Assembler code. These features all work rather well. One of the nicest things about the Lucidata Pascal package, is the large number of demonstration programs supplied in Pascal source form so that you can see by reading working code, just how the various features are used. I have one small bit of advice. If you buy Lucidata Pascal and you find the instructions for using some extension a bit vague, consult the "syntax diagrams" in the back of the manual. They will usually clarify the syntax.

I had some mysterious trouble when I tried using External assembler code, and I should mention the reason

for it. I put my code up in high memory, well above where I knew Lucidata's program and stacks would be located. It turns out that the runtime package checks for live memory by modifying the first byte of every page of memory above its program and stack area, and up to the value stored in FLEX's MEMEND. You must modify MEMEND with your assembler code, by overlaying the value there with a lower value. Part of your external routine assembler code should therefore be:

```
ORG MEMEND
FDB $B000
```

Of course you may move MEMEND down as far as you like, but with the above, it would be safe to put your assembler code anywhere from $B000 to $BFFF.

ABASIC

I have in hand at this writing a preliminary copy of ABASIC for the 6809 from Frank Hogg Laboratories. Most of it works perfectly. It is very efficient as a compiler for anyone not needing floating point math. ABASIC has excellent string handling capabilities, and it is very good for writing programs to "filter" text files. A long time ago, I wrote myself a BASIC compiler using the 6800 version of ABASIC (actually more of a text file preprocessor) that accepts a source text with labels and long variable names, and outputs a file compatible with either ABASIC or TSC BASIC. ABASIC differs from most BASIC's in that line numbers are only required if they are the "target" of a GOTO or a THEN. That is, any line that is not the entry point for a branch, needs no line number. If I had one wish concerning ABASIC, it would be that reasonably long labels could be used in place of line numbers, and that variable names could be longer than one character and one digit.

ABASIC formats the source text as it processes it. The code produced is very efficient, and modules are only included if they are called. The claim of "no runtime package" is technically correct, though some code modules are generated that are subroutines for such things as file handling, I/O, etc. In the sense that they are included as part of the compiler output and don't require the loading of a separate set of runtime routines, there is no runtime package.

The problems remaining, are involved with the output formatting with comma and semicolon in Print statements. I am expecting a copy with these problems cured, very shortly.

Write 'N Spell

Peter Stark has expanded the capabilities of Spell 'N Fix to include the facility to look up a word in the dictionary while editing a file with the TSC editor. If you use your system for word processing regularly, you may be very interested in this new package. See the ad for Star Kits for prices.

MAC95 Macro Preprocessor

MAC Software, P.O. Box 1129, Deluth GA 30136, has sent me a copy of their MAC95 software. Macros that you write for this package are not like those written for a Macro Assembler. A Macro Assembler will include in line, the code for the macro each time you call it. This Preprocessor generates the code necessary to push the parameters to be passed to a macro subroutine, at each call of the Macro. The macro itself is included as a subroutine, and in it, is generated the code to handle the parameters and "fix" the stack. The main idea is sort of "to raise the level of assembler programming" to something that begins to look like a higher level language.

The effect is to add new instructions to the 6809 instruction set. It is sort of a "design it yourself" high level language. This kind of thing is ideal for generating very efficient code and developing a language tailored to a specific application. The package looks very well done, but I have not had a chance to do much more than read the instruction manual once over lightly. I will report more in a future column.

FORTH for the Color Computer

I spent a couple of months a year or so ago, playing with FORTH, and becoming familiar with it. I've reported at length on the language in a recent issue of '68' in which I indicated the similarities and differences between some of the most common languages for Microcomputers. XFORTH, a previous product of Frank Hogg Laboratories, is a version of FORTH that has been prepared specially to be FLEX compatible. I've reported that XFORTH had the best manual I'd ever seen on FORTH.

This present package is a version of FORTH tailored to the Color Computer. This version is Radio Shack DOS compatible. That is, it runs directly without FLEX. It has an excellent manual also. It is sort of a "digest" of the larger manual for XFORTH.

I have run it briefly, after a look at the instruction manual to refresh my memory and find out how to get it running. The main difference between this version and standard FORTH is that the unit of disk memory (a SCREEN in FORTH) has been juggled to be 32 lines of 32 characters rather than the standard FORTH screen of 16 lines of 64 characters. Of course this has been done to make the screen format compatible with the CC. As I've mentioned before, FORTH is not just a language, but a whole operating system. There are two Editors included with it, a simple line editor, and a full "Screen Editor" (one in which you move the cursor around the screen and change characters or insert words anywhere.) CC FORTH also includes a full Assembler.

I have to admit that on my personal preference list for languages, FORTH is on the bottom. I've said many times before that we don't all have to have the same tastes. Indeed it would be a dull world if we were all that uniform. After all, it is the final results that count. If you think you might like FORTH based on my previous descriptions, and you have a Color Computer, this might be just what you are looking for. If you are tired of working with the "conventional languages" and want to try something that will let you do the most with the least code, FORTH may be what you ought to try. See the ad for Frank Hogg Laboratories in this issue.

MULTIPLY PROGRAM

I realize that I've gotten off schedule with the integer math program project we started several columns ago. It seemed as though the items for review were piling up, and I thought we had better cover as many as possible this time. I promise to get back to that project next time.

REVIEWS

Product reviews are an important and necessary part of 68 MICRO Journal. It is by this method that we determine, in many cases, that a new product merits advertising space in 68 MICRO Journal. You would be surprised at how many "new" products are sent in for advertising o.k. and product review, that do not, for one reason or another, measure up. Many times we are able to furnish hints and suggestions to the manufacturer that enables him (or her) to redo some portion of the product and pass our lab test.

By getting the product "up to snuff" and performing as to be advertised, we afford a certain element of reader protection. We certainly have not been 100

percent successful in this endeavor, but we have come closer than any other computer magazine. I am not going into why we do not want to be "police" again (once a year is enough) but there are some points that I feel need touching upon.

First, if you are a vendor or manufacturer and we have not tested your products before, you **will** have to get our lab approval before the advertising can run in 68 MICRO Journal! Just as it is stated in our advertising "rate sheet". Also we have had a few problems recently with readers contacting vendors and manufacturers wanting a particular item for review, this without our prior knowledge. This has caused some problems with our review policies. Especially where we anticipate a product to be reviewed and have pre-established an agreement with some other party to do the review. This as you can see, can become as they say "a sticky wicket".

We earnestly solicit reviews from any reader on any product that our readers might be interested in. Even products that are not presently and never have been advertised in 68 MICRO Journal. The main thing about reviews is that they alert you, our reader, to the good features of a particular product, or the poor features of a particular product. I **am not** attempting to dictate to you what to buy, I only want you to know as much as you can about what you are considering laying down your hard earned cash for. Nothing more. Also I think that all of you understand that just because a product is not advertised in 68 MICRO Journal that it is not a "bum" product. Quite the contrary, there are some excellent products, advertised in other media, that have never been advertised in 68 MICRO Journal.

If you want to do a review, and one has not been done before, please contact me and we will try to arrange it, if it has not already been scheduled, either in-house or with the vendor or manufacturer. But I have come to the point that I will **not** support a request for a product that we have no prior knowledge of! Actually this recently occured, I refused to honor the request and found out later in the day that the person requesting the product was actually the brother-in-law of the vendor. To make matters worse we ordered in the product (paid for it with order) and found that it was full of "bugs" when we tested it in-house. I demanded our money back from the vendor and was told that, "well it does most everything, and we will get the patches out to all customers, as we get them done". This led to a rather heated discussion, and some pointed suggestions, on my part as to how it could be made right. Well, believe it or not, but they took my suggestions, sent me a "review" copy this week (already refunded our money) and by golly, it works just dandy. Everybody profits. Should be advertised next month or so and reviewed around the first of the year.

On the whole we have had little problem with our advertising approval policies. Many advertisers have said they appreciate the way we do it. It **does** add a degree of "faith in a product" if we review or test it and give it a passing grade. Many advertisers tell me that sales **really** take off after a good review. Actually I allow no poor reviews as there just is no space for that type thing. And if it is bad enough not to pass lab testing then it will not be advertised, so no wasted space on reviews of products you will not likely be buying. And where we or the vendor have slipped up and let a "poor" product be advertised, we **stopped** the advertisement and immediately told you about it. Remember?

There are a few "bad apples" advertised in other magazines. A few of our readers have purchased these products and have had problems, to some degree or another. These have been "properly" reported to us and we are looking into them. As soon as we receive sufficient, verified, information, I will let you know. Have to be sorta careful on that sort of thing, already had to defend one law suit. We settled it before it got into court and let it go after they paid our legal fees and gave me written agreement that it would not be advertised or sold until they got it working right. I am pickey about our advertisers, I owe you at least that much.

Also while on the subject of reviews there is another point that needs to be covered. That is that we believe that we have received items for review, and have not gotten the reviews done.

There are several reasons why this would happen. First, about a year ago we moved our entire operation. As you can well imagine we misplaced some review items. During the following months we thought that we had "dug out" most of those products and started them down the road towards review. Then last month we were flooded. Our entire building was flooded with about 6 inches or so of water during a "100 year flood", whatever that is. Well in moving things off the floor (in the middle of the night and with no electricity) some items may have again been misplaced.

If you have sent us a product for review, and after a reasonable length of time (60-120 days) have heard nothing, please get in contact with me. Either we have misplaced the item or maybe even did not get it. We have received items, in the mail, that were mailed months before and wandered all over, before finally getting delivered. Also I strongly suspect that there are some items that we will never receive. We get letters quite often from new subscribers, stating that they had sent in a new subscription with a check, and have not received the magazine and the check has never cleared their bank. Well, you can certainly believe me, if we received the check, we sure would have put it through the bank. This leads me to believe that some items of mail we never receive. Items for review could just as easily become non-items.

Survey after survey always indicates that product reviews are one of, if not the most popular type of articles we run. Advertisers tell us that reviews do more to move their product than any one other thing, including the advertising itself. Reviews can tell you the good, bad and indifferent aspects of a product. Reviews can in many instances give a new advertiser a certain amount of immediate acceptance. Reviews can fill in between the lines of advertising and the knowledge that comes from "hands on" experience. Reviews serve well both the seller and the buyer, they are the bridge that conveys trust. For these reasons, we also feel that reviews are one of, if not the, most important articles we publish.

So, if you have sent something in for review and have not heard anything on it, **PLEASE** let me know. If you are using a product that you think that the readers of 68 MICRO Journal should or would want to read about, **please** let me know. It does not have to be a full article, a letter will do. Just make sure that it is typed (dark) ribbon, or on a FLEX disk (disk will be returned). See page two (2) for formats. We all will thank you.

DMW - - -

DATA SYSTEMS Dual Serial Card

Recently received for review was a dual serial card from a fairly new advertiser. Data Systems 68, 2316 Diversified Way, Orlando, Fl. 23804. With all "new" advertisers we are somewhat more detail in our initial inspection of their product. Having been "bitten" once or twice we try to screen advertisers more so than in the past. Needless to say this has made some unhappy. But I feel that it is much better than letting our readers get stung. After testing this product I felt that you might like to know about a company that seems to know what they are doing.

Actually what can be said about another serial card? Well in this case, we found not only a well designed and laid out circuit board but a novel twist as well.

First, a look at the card. The card is both silk screened and solder masked. Baud rate selections are from 110 to 9600 baud, with strapping for IRQ, NMI and IRQ-2 for each ACIA (MC6850).

The two ACIA's have their addressing decoded by a simple inverter (CS1 lines). The TX data, RTS, RX data, DCD and CTS lines are driven to true RS-232 by 1488 and 1489 drivers. The 5v is regulated by a standard 7805, the + and - 12v lines are derived from the Standard S30 Bus and zener regulated.

The "novel twist" mentioned above is a provision for the addition of a transistor to trigger the interrupt line. With some software this could easily allow single stepping or other interrupt type programs. A SPDT switch could be hooked to a port, on the card, and the furnished "debounce" circuit and the above transistor would drive the NMI line.

The instructions come complete with circuit drawings, parts overlay (as well as silk screening). Also mentioned in the instructions is some interesting information for SWTPC SWTBUG™ users.

Seems that SWTBUG views a dual serial port of this type as a PIA. So, if this card is to be used on ports 0 or 1 then the following changes should be made to SWTBUG:

ADDRESS	OLD	NEW
$E1FA	$28	$00
$E211	$2E	$00

The above is necessary as SWTBUG test for port types and would confuse the board for a PIA.

Data System 68 also sells the necessary parts to completely populate the board. For $19.50 unpopulated the dual serial board is a good buy.

A 68 Micro Journal review - - -

MUSIC BOARD

Review of Palm Beach Software Music Board

I had better preface this by saying that I have been interested in music for a long time. Several years ago I built an electronic organ (my own design) using TTL digital frequency dividers, and Integrated Circuit Operational Amplifiers. I have never before gotten interested in "computer music" because I felt it to be too "mechanical".

When my Music Board arrived, I read the instructions carefully, and gingerly plugged the board into my old SWTPC in slot 3. I then connected the RCA plug cable to my stereo amplifier that happens to be in the same room as my computer. The software is set up for the newer systems with 16 I/O addresses per port, and a 2 MHZ clock. With the help of the instructions it was no trick to convert the programs to run on my system, and I tried running one of the 25 song files on the disk.

I must say that I was a bit taken aback by the quality of what I heard coming from my old AR-2 speakers. It was not quite what I had expected to hear. There was four part music with a number of different and distinct "voices" coming at me. I listened for several minutes and then went and disconnected my rotating speaker from the organ and connected it to one channel of the stereo amplifier. My small rotating speaker is not very good for low notes, but it is made to "spray" the highs around the room. It made such a big difference in the organ that I thought it might enhance the computer music too, which it indeed did.

I realize that not many computer users happen to have an electronic organ in the house, but if you do, and can run the sound through your rotating speakers, you will note that the music sounds remarkably like a "player organ".

The package includes MOZART, a music editor. With it, I was able to enter the verse portion of The Sound of Music, in about 45 minutes. Nobody claims that it is easy to enter four part music, but the results are very satisfying. MOZART is a great help, since you may at any time play what you have entered to that point, for a check on progress. You may delete notes, insert notes, change chords, tempo, duration, or voices. You may also repeat any portion of the song. MOZART prompts for the first and last note to be repeated.

The instructions (and MOZART) refer to the four parts of the music (four notes at a time for a chord of four parts) as the four voices. Voice 1 is generally melody, while voice 4 is the Bass or Pedal note. The "timbre" or quality of the notes may be changed by selecting from a table of 5 different waveforms. Voices 1, 2, and 3 may be played with any of he five waveforms selected independently for each, while the Bass voice uses only waveform 3, the Sine wave. Each waveform table is a 256 bytes long, and you may assemble your own to overlay the ones supplied. An interesting variety of waveforms is supplied. The most bland is a pure sine wave. Next is a triangle wave, a sine wave with minimal harmonic content. Two others were made by summing sine waves at the fundamental frequency and a couple of harmonic frequencies. One sounds rather like a horn and is composed of fundamental, second, and third harmonics, in undisclosed ratios. The second contains fundamental, third, and fifth harmonic, and sounds rather like a flute. The last is a square wave, which is far too raspy for my taste.

After listening to all the selections on the disk, which plays for over an hour, I decided to try my hand at waveforms, and I wrote a BASIC program to generate a table, given the function desired. I spent an evening playing with decaying sine waves, sawtooth waves with low pass filtering, pulses, square waves filtered, etc. and I think I came up with some interesting sounds. I had been through much of that exercise when I built the organ.

The next night, however, I really had a good time. I modified that BASIC program to accept relative amounts for the fundamental and harmonics through the 10th, and generate a table for the sum of all of the components. Specifying the amounts of the various harmonics is exactly what is done by the "drawbars" on a Hammond organ. I was able to make a passable STRING stop, a HORN, an OBOE, a STOPPED FLUTE, a DIAPASON, and several others. The range of sounds was extended considerably beyond what was originally supplied. I've sent the waveform generator program and some of my results to Palm Beach Software, since the manual requested that users generate some further waveforms. I also mentioned the fact that it would be nice to be able to select from more than 5 waveform tables.

The disk has a dozen Christmas Carols, a selection of popular music (no rock), and some classical selections. The variety really shows off the music card. Many of the selections are very well done arrangements. A few are mediocre, probably having been done by someone with a small speaker with no low frequency response, so he coded the BASS part an octave too high. That could be fixed with MOZART, and I intend to improve some of the arrangements. One or two are just plain awful! The Good to Excellent ones greatly outnumber the one or two bad ones, however.

MOZART has provision for inserting a very brief pause between notes repeated at the same pitch. Rather than coding a quarter note, you use a code for a 5th note, etc. The result is to leave a 64th note pause so that the repeated note can be heard as such. I did find that there is no provision for "triplets". 12th notes would work nicely for that function, since a note can be repeated without a break. A quarter note triplet is a group of three notes that are to be played equally and in

the same time as two quarter notes are played in another voice of the piece. Therefore, the triplets would be 2 12th notes each and the quarter notes 3 12th notes. That is not a terribly serious omission, and I suspect that such a provision will be added later.
The most serious omission from the package, is the lack of any sort of expression control. The loudness or volume is not programmable in any way. Tempo is programmable, and with care could be used to program a retard or tempo change rather smoothly by changing the tempo a tiny bit between notes or chords, though very few of the songs on the disk have used tempo changes effectively. Have I mentioned the fact that Source Code is provided for all the software?
I thought about the lack of loudness or expression control, and decided that the MUL instruction could easily be used to multiply the waveform table sum by a loudness value before outputting it to the D/A converter. A long evening had that working and a patch job on the compiler so I could include a loudness parameter in a line where TEMPO is specified. It works reasonably well, and I will send my patched up version and a sample song to Palm Beach Software. I figure that if they like it, they will do something to include the loudness parameter more cleanly than my patch to show that it does work. My modification causes no hardware change, and is compatible with old music that doesn't contain loudness parameters.
The board is simple and cleanly designed. Noise from the computer is not a problem in the audio output. A cable is supplied, which runs to a panel section with a level adjustment and the RCA phono jack. The panel is meant to fasten into the back of the newer SWTP and GIMIX chassis, but it was no trouble to run the cable out through one of the holes in my antique SWTPC system box. If you are getting the idea that I am pleased with this hardware software combination, you are quite correct. I've just now been able to tear myself away from playing with it long enough to write this review. At $75 for the board and a disk full of music to play, this has to be a very good buy for anyone interested in trying something different with his computer. I have to quit now and go back and code some more music.

Additional Note (no pun intended)
I just received a call from Dan Farnsworth of Palm Beach Software, in response to a letter in which I mentioned the lack of triplet capability. He gave me the patches over the phone and promised me a patched copy some time in the near future. Dan also outlined the simple way to allow my choice of 13 or 29 waveform tables. That puts some real variety into the music, being able to create a bunch of "solo" waveforms and "accompaniment" waveforms, and having the ability to select a different one for each voice in the music!

To order this board and software, see Palm Beach Software ad in this issue.

Ron Anderson

COLOR User Notes

ROBERT L. NAY
4429 Plantation Lane
Norcross, GA 30071

This month we will look at another Game from Computerware, an excellent FORTH implementation on the Color Computer, and a Cooling Fan for the Color Computer from Atomtronics. We will also look into the "Full Functioning Keyboard" for the Color Computer.

EL DIABLERO -- An Adventure Game
COMPUTERWARE
Dept C, Box 668
Encinitas, Ca. 92024

One of the GREAT things about the Color Computer is that you can relax with one of any number of different Games after spending long hours sweating over a "Working Program". Another excellent one was received from Computerware the other day, and we have really enjoyed "Working" with it the last couple of weeks.

As you are probably aware, Adventure Games have many different themes; El Diablero is a "witchcraft", or "Magic", version of the series. It is a Machine Language program that uses up almost all of the RAM in a 16K Color Computer (and takes a while to Load from Tape), so its' reaction to an input is fairly quick. It is based on the standard two word command line of a verb and noun, such as "GO WEST", but is optimized for the Color Computer by allowing the use of the Arrow Keys for direction movement. This is a nice feature, as typing "GO WEST" numerous times gets old. It also provides for saving a game in progress, IF YOU ARE NOT IN immediate DANGER, so you can easily pick up where you left off when you got "stumped" and decided to "sleep on it a while". You can also purchase a set of maps and hints for $5.

Your "situation" in El Diablero is that you wake up, dazed and confused, in the middle of a desert in the southwest, knowing that you had been learning the techniques of sorcery from an old man in these parts. The problem is that you can't remember anything you had learned, and you can't find the old man. You must learn to acquire and use this magical "power" in order to to tackle El Diablero. From here on, GOOD LUCK, you will need it! Another excellent game from Computerware.

COLOR-FORTH -- A FORTH Language on the CC
HOYT STEARNS ELECTRONICS
4131 E Cannon Dr.
Phoenix, Az. 85028
RAM Version -- $58.95
ROM Version -- $123.00

Of all of the "High Level" programming languages, FORTH has precipitated far and away the most controversy. I find it intriguing, and have read about everything I can get hold of on it. I agree with everything Ron Anderson said in his "FLEX Users Notes" column in the Sept '82 Issue of '68' Micro Journal. I have yet to read Brodies' "Starting FORTH" (available from several sources, including the Forth Interest Group, PO Box 1105, San Carlos, Ca 94070 -- $16 in paperback), but I do recommend Thom Hogans' "Discover FORTH" (Osborne/McGraw-Hill, 630 Bancroft Way, Berkeley, Ca 94710) as an easy to read introduction to the language that maintains an easy-going presentation of FORTH that does not get bogged down in the details. If you are 'investigating' the language. It is the book to read.

The RAM Version of COLOR-FORTH comes on Tape with instructions for installing the system on Disk. These instructions are very specific, and if you follow them VERBATIM, you will have no problems and can proceed "FORTH-rightly" onward (am I catching the bug, too??). Stearns' COLOR-FORTH is a FIG-Forth version that has been developed SPECIFICALLY for the Color Computer. It was written to take advantage of many of the unique aspects of the Color Computer, and includes a special "Semigraphic-8" editor which fully utilizes the Memory Mapped Color Display (for example, the Editor provides a Green background if you are in the Decimal Mode, or an Orange background if you are using HEX, Binary, etc.). Another unique feature of COLOR-FORTH is that a TRACE utility has been written, which provides the number of items on the data stack, the top two items on the data stack, the "word" that will next execute, and the whole line is indented according to the nesting level. This is EXTREMELY nice while you are learning to program in

FORTH, to say nothing about it's value in debugging a program.

The Reference Manual that comes with the package is EXCELLENT; I can assure you that MANY hours went into its' preparation. It is not a tutorial on FORTH Programming (as Hoyt states, there are several books available for that - and he provides some recommendations), but it concentrates on describing the unique features of COLOR-FORTH. It begins with the above mentioned Disk Conversion Instructions, and then contains a couple of pages on "Trying FORTH" (this is an excellent approach - you have just obtained a new "toy", and you want to see it WORK - make it say something in "FORTH", before you get "serious"). Next is a page listing all of the FORTH Words provided in the order that they are in the "dictionary", followed by a page listing them alphabetically (all 269 of them). Don't let the long list scare you off; start out easy, a few at the time, and it's not that bad. Besides, that is a lot of "Words" you won't need to define yourself. Then begins a discussion of the COLOR-FORTH Words that are uncommon, interesting, not in FIG-Forth or the Forth 79 Standard, or are completely unique to the Color Computer; alphabetically. In most cases, the discussion indicates how to change the "standard" Word to a COLOR-FORTH Word or statement. The alphabetic listing makes this conversion MUCH easier than it would normally be. Next is a discussion of Words that are not in COLOR-FORTH, but are common in other Forths; this is a tremendous aid if you are converting a different Forth Program to run with COLOR-FORTH - again, alphabetically. Next is about 13 pages of the "fig-FORTH GLOSSARY". Then you get some listings of odds and ends Hoyt has provided, such as Conway's game of LIFE, a random number generator, implementations of PSET and PCLS in PMODE 4, the disk and tape copy utilities, and the program that was used to provide the previously mentioned sorted vocabulary list (by the way, all of these listings take up 1 and 1/2 pages - interesting - how many pages would it take to list them if they were written in BASIC??). These are followed by the Memory Map, COLOR-FORTH System Variable Table, and a list of the FORTH Extensions provided. Next is the COLOR-FORTH Semigraphic-8 Screen Editor Manual, followed by the instructions for installing the COLOR-FORTH ROM in the Disk Controller Cartridge (I didn't mention that, did I? - It doesn't affect the Disk Operations and provision is also provided for installing a Game ROM in place of the Disk ROM, if you like). Finally, you get a FIG (Forth Interest Group) Mail Order Form and a handy "FORTH Handy Reference" Sheet (I said it was 'handy'). All in all, there is about 70 pages of information provided for your "edification" and study.

This is the first FORTH Implementation I have seen for the Color Computer, but I am impressed. HOYT STEARNS ELECTRONICS has done an outstanding job of providing an EXCEPTIONAL piece of Software for the Color Computer, and the Reference Manual could be used as a model for a complete presentation of a piece of Software. It is not a complete "stand alone" FORTH Programming Manual, but neither is any other Manual I have seen for a "Programming Language" piece of Software; but it sure is an adequate presentation of the Implementation of the Language for this Computer System. If the FORTH Programming Language interests you, this package will be hard to beat.

COLOR FAN - A Cooling Fan for the Color Computer
ATOMIC CITY ELECTRONICS (Atomtronics)
3195 Arizona Ave.
Los Alamos, Nm. 87544
with mounting hardware -- $34.95

We have looked at several of the Atomtronics Products in the past, and now have another one from them. Most Users who have been following this Column are aware of the Power Supply and Heat problems of the Color Computer, and have seen several of the published 'fixes'. Atomtronics has now come out with a 3" COOLING FAN for the Color Computer which is easy to install and will help alleviate some of these problems.

Installation consists of removing the Top Cover and Keyboard of the Computer and mounting the Fan under the left side of the Keyboard over the Air Slots in the bottom case. These slots are opened up, and a pair of leads are run to the AC Power Leads at the Power Transformer. Reassemble the Computer; Installation complete. Simple!! Remember one thing, though. When you turn the Computer OFF with the Power Switch at the back, YOU DO NOT DISCONNECT AC POWER from the Computer. This means you MUST DISCONNECT THE AC POWER CORD, UNplug it, ANY TIME you are going to work on it. Secondly, the COLOR FAN will NOT be turned off when you turn the Computer off. I STRONGLY recommend, as does Mike Wolf, who designed this modification, that you connect Power to the Computer through a Multiple Outlet that has a Power Switch so you can turn everything OFF at ONE TIME, which REMOVES ALL AC POWER FROM THE COMPUTER. My procedure is to turn the Computer off with its' Power Switch, and then turn the Multiple Outlet Switch OFF, shutting down the Printer, Disk Drives, Computer, etc. Turn-on consists of turning the Multiple Outlet ON, and then the Computer Power Switch.

This is a product that we have needed for the Color Computer for a long time, and will greatly extend the life of the parts and pieces in the Computer. The Installation is simple, and the Price is excellent. Thanks again, Mike.

PS: I have mentioned the Atomtronics DISC BOARD several times in the past (available from DATA-COMP), and discussed its' capabilities. We have been working with Mike to get this unit "on-line" for several months now, and it is really looking good. For those who may have missed it, it is a Board that measures about 10" sq and includes a 5 1/4" Disk Controller, a Real Time battery maintained Clock with an interrupt timer, a Parallel Printer driver, and an optional MC6845 controlled 80x24 Video Display option for use with video monitors. It has a connector to the ROM Pack Slot, but still allows other ROM Packs to be plugged in with it. There are several units in the field at the present time, and the unit is working very well. The only thing left to get up and running is the Double Density Software, which should be along shortly. I intend to give you a detailed description of the DISK BOARD next month.

Another item from Atomtronics to watch for in the near future is a Communications Board which uses the MC6850 ACIA and includes the full 128 ASCII character set, single key Printer Control, etc. This unit is made especially for those Users who have access to a large time shared Computer System, such as is used by many of the Colleges and Universities. Mike has submitted the Keyboard Driver that he developed for this package for your "edification".

I have had discussions with several of the people who have been developing Software for the Color Computer over the past several months concerning all of the different "methods" being used to obtain the full ASCII Character set with the Color Computers' Keyboard. Generally, most systems use the <CLEAR> Key as a CONTROL Key, but there the similarities end. There is basically two different approaches to the problem: one is to leave as many of the Key Functions of the Color Computer "as is" as possible, figuring that most of the Users are already comfortable and used to that layout; the second is to change them as much as is feasible to approach

the normal layout of most Terminals. I have had quite a bit of experience with BOTH Keyboards, and feel that the "feel" of the Color Computers' Keyboard is different enough from that of the "normal" Keyboards to make the conversion, in use, fairly easy, so I lean towards leaving as many of the Key functions of the Color Computer alone as possible. For instance, it is not hard to get used to hitting the <CLEAR> Key for CONTROL on the Color Computer, and then use the normal CONTROL Key on another Terminal (which is not THAT standard, by the way). Normally, I don't see any real problem with the different Keyboards. An exception is the ESCAPE Key; it is almost always the upper left Key on any Terminal, and is normally used in a slightly different situation than the other Keys (especially when using FLEX). I just use the FLEX TTYSET.CMD to make the Number 1 Key the ESCAPE Key, and don't have any problems. Anyway, we have many different Keyboard "Layouts" in the various Word Processors, Communications Packages, and FLEX Conversions; and would like to start a "Standardization" movement. It would be nice to have every one on the same wavelength, so to speak. Besides providing an example of how the Key Functions are changed via Software on the Color Computer, Mikes Routines provide a suggested Layout for your consideration. Let me know your feelings on the subject, and maybe we can come up with a Standard, Full Functioning Keyboard for the Color Computer as time passes.

--- RLN ---

```
*************************************************************
*
* Radio Shack Color Computer keyboard scan routine.
*
* Written by:   Michael Wolf
*               3195 Arizona Ave.
*               Los Alamos, NM 87544
*
* All rights to this program are turned over to the
* public domain in hopes that it will become the
* standard for an improved keyboard routine for all
* programs requiring the special codes that it pro-
* vides.  This program may be incorporated into any
* program sold for profit without restriction pro-
* vided that credit is given to the author.
*
* It gives you the ability to generate all 128
* ASCII characters and control codes plus 14 ESCAPE
* sequences (ESC 1 through ESC = and ESC A).  The
* key values are contained in a keytable and thus
* may be easilly altered to fit any special appli-
* cations that the user may require.
*
* Entry and exit parameters are identical to the
* routine in Color Basic and thus may be substituted
* without change in any program using POLCAT.
*
* Some of the keys (all arrow keys, CLEAR, @, and
* BREAK) generate codes different than those gen-
* erated by BASIC.  CLEAR functions as the CONTROL
* KEY.  A Shift "@" generates an "`".  A Shift BREAK
* generates a DEL.  And the arrow keys generate ES-
* CAPE sequences except for the Unshifted left arrow
* which generates the standard Backspace Character.
* All of those codes lost are available as CONTROL
* Codes by holding the CLEAR Key and hitting another
* key; for example, to generate the Color Computer
* CLEAR (form feed, which clears the screen), hold
* the CLEAR Key down and hit the L Key (CONTROL L
* is the Form Feed, which clears the Screen on the
* Color Computer).
*
* The keyboard generates the following codes:
*
* key   Unshifted    Shifted      CONTROL   Caps mode
*  @     $40 = @     $60 = `    $00 = nul     $40
*  A      61 = a      41 = A     01 = soh      41
*  B      62 = b      42 = B     02 = stx      42
*  C      63 = c      43 = C     03 = etc      43
*  D      64 = d      44 = D     04 = eot      44
*  E      65 = e      45 = E     05 = enq      45
*  F      66 = f      46 = F     06 = ack      46
*  G      67 = g      47 = G     07 = bel      47
*  H      68 = h      48 = H     08 = bs       48
*  I      69 = i      49 = I     09 = ht       49
*  J      6A = j      4A = J     0A = lf       4A
*  K      6B = k      4B = K     0B = vt       4B
*  L      6C = l      4C = L     0C = ff       4C
*  M      6D = m      4D = M     0D = cr       4D
*  N      6E = n      4E = N     0E = so       4E
*  O      6F = o      4F = O     0F = si       4F
*  P      70 = p      50 = P     10 = dle      50
*  Q      71 = q      51 = Q     11 = dc1      51
*  R      72 = r      52 = R     12 = dc2      52
*  S      73 = s      53 = S     13 = dc3      53
*  T      74 = t      54 = T     14 = dc4      54
*  U      75 = u      55 = U     15 = nak      55
*  V      76 = v      56 = V     16 = syn      56
*  W      77 = w      57 = W     17 = etb      57
*  X      78 = x      58 = X     18 = can      58
*  Y      79 = y      59 = Y     19 = em       59
*  Z      7A = z      5A = Z     1A = sub      5A
*up arr   1B 33=e3    1B 34=e4   1B 35=e5
*dn arr   1B 36=e6    1B 37=e7   1B 38=e8
*lft ar   08=BS       1B 39=e9   1B 3A=e:
*rt arr   1B 3B=e;    1B 3C=e<   1B 3D=e=
*BREAK    03=etc      7F=del     1B = esc
*  0      30 = 0    Caps toggle  1B 31=el
*  1      31 = 1      21 = !     7C = |
*  2      32 = 2      22 = "     7E = ~
*  3      33 = 3      23 = #     5E = ^
*  4      34 = 4      24 = $     1C = fs
*  5      35 = 5      25 = %     1D = gs
*  6      36 = 6      26 = &     1E = rs
*  7      37 = 7      27 = '     1F = vs
*  8      38 = 8      28 = (     5B = [
*  9      39 = 9      29 = )     5D = ]
*  :      3A = :      2A = *     1B 41=eA
*  ;      3B = ;      2B = +     1B 32=e2
*  ,      2C = ,      3C = <     7B = {
*  -      2D = -      3D = =     5F =
*  .      2E = .      3E = >     7D = }
*  /      2F = /      3F = ?     5C = \
*
*************************************************************

0152            KEYSCR  EQU     $0152
FF00            PIA1    EQU     $FF00
011A            CAPLOK  EQU     $011A
03FF            ESCFLG  EQU     $03FF
03FE            RPTCTR  EQU     $03FE
03FD            KEYREG  EQU     $03FD
0088            CURSOR  EQU     $0088

C000                    ORG     $C000
C000 84  3F     ESCAPE  ANDA    #$3F        Strip upper bits
C002 8B  30             ADDA    #$30        Convert to number
C004 B7  03FF           STA     ESCFLG      Into pending register
C007 86  1B             LDA     #$1B        ESCAPE Char.
C009 20  3C             BRA     PC4

C00B B6  03FF   ESCS    LDA     ESCFLG      Get pending char.
C00E 7F  03FF           CLR     ESCFLG      Clear the flag
C011 20  34             BRA     PC4         Return it

*
* Poll keyboard; return A=key, Z flag=0 if key
*  down or A=0, Z=1 if no key pressed.
* The CLEAR Key functions as a second level shift
*  or CONTROL Key.
*

C013 34  14     POLCAT  PSHS    B,X         Save regs
C015 7D  03FF           TST     ESCFLG      ESCAPE pending??
C018 26  F1             BNE     ESCS
C01A 8D  43             BSR     POLCT       Scan keyboard
C01C 24  3D             BCC     PC1         If no key, set Z and rtn
C01E 81  C0             CMPA    #$C0
C020 24  29             BHS     SPEC
```

```
C022 8E C0FD      PC0   LDX   #LOOKTB   Look up real key valu
C025 5F               CLRB
C026 1E 89            EXG   A,B
C028 30 88            LEAX  D,X
C02A A6 84            LDA   X
C02C 81 80            CMPA  #$80      Shift 0??
C02E 26 05            BNE   PC3
C030 73 011A          COM   CAPLOK    Toggle caplok
C033 20 26            BRA   PC1       Return no key
C035 4D           PC3 TSTA
C036 2B C8            BMI   ESCAPE    If it is a special ke
C038 7D 011A          TST   CAPLOK    convert to escape seq
C03B 27 0A            BEQ   PC4       Caps lok on??
C03D 81 61            CMPA  #$61      Is it lower case?
C03F 25 06            BLO   PC4
C041 81 7A            CMPA  #$7A
C043 22 02            BHI   PC4
C045 84 DF            ANDA  #$DF      Conv. to Upper Case
C047 1C FB        PC4 ANDCC #$FB      Clear Z
C049 35 94            PULS  B,X,PC    Rtn to calling prog.

C04B 81 F0        SPEC CMPA #$F0      ENTER?
C04D 26 04             BNE  SPEC1
C04F 86 0D             LDA  #$0D
C051 20 F4             BRA  PC4
C053 81 DF        SPEC1 CMPA #$DF     SPACE?
C055 26 04             BNE  PC1
C057 86 20             LDA  #$20
C059 20 EC             BRA  PC4

C05B 1A 04        PC1 ORCC  #4        Set Z flag
C05D 35 94            PULS  B,X,PC    and return

C05F 32 7D        POLC1 LEAS -3,S     Save operating room
C061 8E 0152          LDX   #KEYSCR   X to scratch regs.
C064 6F E4            CLR   0,S
C066 C6 FE            LDB   #$FE      Set up 1 side of
C068 F7 FF02          STB   PIA1+2    keyboard matrix.
C06B 8D 78        POLC2 BSR   GETCOL    Get value
C06D A7 61            STA   1,S       Save it
C06F A8 84            EORA  0,X       Mask with old data
C071 A4 84            ANDA  0,X
C073 E6 61            LDB   1,S
C075 E7 80            STB   X+        Put new value in keyscr
C077 4D               TSTA
C078 26 0C            BNE   POLC3     If 0, no new key
C07A 6C E4            INC   0,S       Inc row pointer
C07C 53               COMB            Set carry
C07D 79 FF02          ROL   PIA1+2    Scan next row
C080 25 E9            BCS   POLC2     Last row??
C082 1C FE            ANDCC #$FE      Set carry if no new
C084 35 94            PULS  B,X,PC    key and return.

C086 F6 FF02      POLC3 LDB   PIA1+2  Compute value of
C089 E7 62            STB   2,S       the key.
C08B C6 F8            LDB   #$F8
C08D CB 08        POLC4 ADDB  #8      Add 8 for each row
C08F 44               LSRA
C090 24 FB            BCC   POLC4
C092 EB E4            ADDB  0,S       Add coluan value
C094 E7 E4        POLC5 STB   0,S
C096 8E 0454          LDX   #$0454    Debounce delay
C099 8D 5D            BSR   LDELAY
C09B E6 62            LDB   2,S       Recheck
C09D F7 FF02          STB   PIA1+2
C0A0 8D 43        POLC6 BSR   GETCOL
C0A2 A1 61            CMPA  1,S       Same??
C0A4 35 02            PULS  A         Get the value
C0A6 27 04            BEQ   CHECK     If diff, must be
C0A8 1C FE        POLC7 ANDCC #$FE    bounce, so
C0AA 35 90            PULS  X,PC      Carry and return.
C0AC 81 31        CHECK CMPA  #$31    CONTROL Key??
C0AE 27 F8            BEQ   POLC7
C0B0 81 38            CMPA  #$38      SHIFT Key??
C0B2 27 F4            BEQ   POLC7

                  *
                  * RETURN and SPACE are special cases --
                  *
C0B4 81 30            CMPA  #$30      RETURN??
C0B6 27 1E            BEQ   SPC
C0B8 81 1F            CMPA  #$1F      SPACE??
C0BA 27 1A            BEQ   SPC
C0BC 81 32            CMPA  #$32      BREAK??
C0BE 26 02            BNE   CK1
C0C0 86 1F            LDA   #$1F      Move BREAK into matrix
C0C2 8D 16        CK1 BSR   CKSHFT    Check for shift
C0C4 26 04            BNE   CK2
C0C6 8B 30            ADDA  #$30
C0C8 20 08            BRA   CK1

C0CA C6 FD        CK2 LDB   #$FD      Check for CONTROL
C0CC 8D 0E            BSR   CKCON
C0CE 26 02            BNE   CK1
C0D0 8B 60            ADDA  #$60
C0D2 1A 01        CK1 ORCC  #1        Set carry if valid key
C0D4 35 90            PULS  X,PC      Restore stack and rtn.

C0D6 8B C0        SPC ADDA  #$C0      Set MS bits if a
C0D8 20 FB            BRA   CK1       special code.

                  *
                  * Check for SHIFT or CONTROL Key --
                  *
C0DA C6 7F        CKSHFT LDB  #$7F    Check for SHIFT Key
C0DC F7 FF02      CKCON  STB  PIA1+2
C0DF F7 FF00           STB  PIA1
C0E2 C4 40             ANDB #$40
C0E4 39                RTS

C0E5 34 04        GETCOL PSHS B       Check coluan
C0E7 B6 FF00           LDA  PIA1
C0EA 8A 80             ORA  #$80      Set MSB
C0EC F6 FF02           LDB  PIA1+2    Check for CONT. or SHIFT
C0EF 53                COMB           Invert
C0F0 C5 82             BITB #$82
C0F2 27 02             BEQ  POLC8
C0F4 8A C0             ORA  #$C0      Ignore SHIFT and CONTROL
C0F6 35 84        POLC8 PULS B,PC

C0F8 30 1F        LDELAY LEAX -1,X    Long delay, enter with
C0FA 26 FC             BNE  LDELAY    delay value in X
C0FC 39                RTS

                  *
                  *  TABLE OF KEYVALUES -------
                  *
                  *  Unshifted, non-control
                  *
C0FD 40 61 62 63  LOOKTB FCB  $40,$61,$62,$63,$64,$65,$66,$67
C101 64 65 66 67
C105 68 69 6A 6B       FCB  $68,$69,$6A,$6B,$6C,$6D,$6E,$6F
C109 6C 6D 6E 6F
C10D 70 71 72 73       FCB  $70,$71,$72,$73,$74,$75,$76,$77
C111 74 75 76 77
C115 78 79 7A 83       FCB  $78,$79,$7A,$83,$86,$08,$0B,$03
C119 86 08 0B 03
C11D 30 31 32 33       FCB  $30,$31,$32,$33,$34,$35,$36,$37
C121 34 35 36 37
C125 38 39 3A 3B       FCB  $38,$39,$3A,$3B,$2C,$2D,$2E,$2F
C129 2C 2D 2E 2F

                  *
                  *  Shifted
                  *
C12D 60 41 42 43       FCB  $60,$41,$42,$43,$44,$45,$46,$47
C131 44 45 46 47
C135 48 49 4A 4B       FCB  $48,$49,$4A,$4B,$4C,$4D,$4E,$4F
C139 4C 4D 4E 4F
C13D 50 51 52 53       FCB  $50,$51,$52,$53,$54,$55,$56,$57
C141 54 55 56 57
C145 58 59 5A 84       FCB  $58,$59,$5A,$84,$87,$89,$8C,$7F
C149 87 89 8C 7F
C14D 80 21 22 23       FCB  $80,$21,$22,$23,$24,$25,$26,$27
C151 24 25 26 27
C155 28 29 2A 2B       FCB  $28,$29,$2A,$2B,$3C,$3D,$3E,$3F
C159 3C 3D 3E 3F

                  *
                  *  Control
                  *
C15D 00 01 02 03       FCB  $00,$01,$02,$03,$04,$05,$06,$07
C161 04 05 06 07
```

```
C165 08 09 0A 0B          FCB   $08,$09,$0A,$0B,$0C,$0D,$0E,$0F
C169 0C 0D 0E 0F
C16D 10 11 12 13          FCB   $10,$11,$12,$13,$14,$15,$16,$17
C171 14 15 16 17
C175 18 19 1A 85          FCB   $18,$19,$1A,$85,$8B,$BA,$BD,$1B
C179 88 BA BD 1B
C17D 81 7C 7E 5E          FCB   $81,$7C,$7E,$5E,$1C,$1D,$1E,$1F
C181 1C 1D 1E 1F
C185 5B 5D 91 82          FCB   $5B,$5D,$91,$82,$7B,$5F,$7D,$5C
C189 7B 5F 7D 5C

                  *
                  * This is a POLCAT driver that generates a nondes
                  *  structive cursor and gives an auto repeat
                  *  function on all keys.
                  *
                  * INPUT A KEY AND DISPLAY IT
                  *

C18D 8D   04      INCH  BSR   INCHNE
C18F 6E   9F A002 OUTCH JMP   [$A002]

                  *
                  * Input a character from the keyboard into A.
                  *  Preserves all registers except A and gives a
                  *  flashing reverse video cursor.
                  *

C193 36   10      INCHNE PSHU  X
C195 7D   03FE          TST   RPTCTR
C198 26   05            BNE   IN1         If not a repeat, set the
C19A 86   05            LDA   #5          delay to first -
C19C B7   03FE          STA   RPTCTR
C19F 9E   88      IN1   LDX   CURSOR      Get char. position
C1A1 A6   84            LDA   ,X          Get char at that loc.
C1A3 36   06            PSHU  A,B         Save it and B reg.
C1A5 17   FE6B          LBSR  POLCAT      Check keyboard
C1A8 27   11            BEQ   INKEY1      If no key, flash cur.
C1AA 7F   03FE  INKY01  CLR   RPTCTR
C1AD 37   04    INKY0   PULU  B           Restore current cur. val
C1AF E7   84            STB   0,X
C1B1 37   14            PULU  B,X         Restore B,X
C1B3 84   7F            ANDA  #$7F        Clear MSB of kbd value
C1B5 39                 RTS

C1B6 86   05      INKEYX LDA  #5
C1B8 B7   03FE          STA   RPTCTR      Reset counter to nokey
C1BB C6   4F      INKEY1 LDB  #$4F        Set cursor flash rate
C1BD 8D   78      INKEY2 BSR  DEL2B       Delay
C1BF 17   FE51          LBSR  POLCAT      Check keyboard
C1C2 26   E6            BNE   INKY01      If key return
C1C4 5A                 DECB              Dec flash counter
C1C5 26   F6            BNE   INKEY2
C1C7 A6   84            LDA   0,X         Get char at cursor
C1C9 B8   40            EORA  #$40        Reverse the video
C1CB A7   84            STA   0,X
C1CD 7A   03FE  INKEY8  DEC   RPTCTR
C1D0 26   EB            BNE   INKEY2      Go delay again
C1D2 B6   01            LDA   #1
C1D4 B7   03FE          STA   RPTCTR      Reset repeat ctr to fast
C1D7 34   10            PSHS  X
C1D9 86   FF            LDA   #$FF        Reset keydown flags
C1DB 8E   0152          LDX   #KEYSCR
C1DE A7   80    INKEY5  STA   ,X+
C1E0 8C   015A          CMPX  #KEYSCR+8
C1E3 26   F9            BNE   INKEY5
C1E5 35   10            PULS  X
C1E7 17   FE29          LBSR  POLCAT      Check for key still down
C1EA 27   CA            BEQ   INKEYX      If not, loop again
C1EC 20   BF            BRA   INKY0       If key, then repeat it

                  *
                  *  Check status of keyboard; returns a Z flag if
                  *   no new key is down.
                  *

C1EE 34   12      STAT  PSHS  A,X         Save regs
C1F0 8E   0152          LDX   #KEYSCR     X=KEYMAP
C1F3 B6   FE            LDA   #$FE
C1F5 B7   FF02          STA   PIA1+2      Set up scan
C1F8 17   FEEA          LBSR  GETCOL
C1FB 81   FF            CMPA  #$FF        No key in that colum?
C1FD 27   1D            BEQ   NOKEY
C1FF A8   84            EORA  0,X         Mask with old data
C201 A4   84            ANDA  0,X
C203 30   01    STAT2   LEAX  1,X         Bump to next colum
C205 4D                 TSTA
C206 26   10            BNE   STAT3       If not zero, found a key
C208 1A   01            ORCC  #1          Set carry
C20A 79   FF02          ROL   PIA1+2      Look at next colum
C20D 25   E9            BCS   STAT1       If no carry we're done
C20F 7D   03FF          TST   ESCFLG      Sequence key pending
C212 26   04            BNE   STAT3
C214 1A   04            ORCC  #4          Set Z
C216 35   92            PULS  A,X,PC

C218 1C   FB    STAT3   ANDCC #$FB        Clear Z
C21A 35   92            PULS  A,X,PC

C21C A7   84    NOKEY   STA   ,X
C21E 4F                 CLRA
C21F 20   E2            BRA   STAT2

                  *
                  * This is a simpler status check and getchar routine
                  *  STAT2 must always be called before GETKEY.  If
                  *  STAT2 is called twice before reading the key,
                  *  it will be lost.
                  *

C221 34   02    STAT2   PSHS  A
C223 17   FDED          LBSR  POLCAT
C226 27   07            BEQ   SS1
C228 B7   03FD          STA   KEYREG
C22B 1C   FB            ANDCC #$FB        Clear Z
C22D 35   82            PULS  A,PC

C22F 1A   04    SS1     ORCC  #4          Set Z
C231 35   82            PULS  A,PC

C233 B6   03FD  GETKEY  LDA   KEYREG
C236 39                 RTS

C237 17   0000  DEL2B   LBSR  DEL14       28 microsec delay
C23A 17   0000  DEL14   LBSR  DEL
C23D 39         DEL     RTS

                  *
                  * This test routine polls the keyboard and displays
                  *  the HEX value of any key al it is entered.
                  *

C23E 1A   50    TEST    ORCC  #$50
C240 10CE 03FB          LDS   #$03FB
C244 17   FF4C  TEST1   LBSR  INCHNE
C247 BD   1B            BSR   OUTHA
C249 B6   20            LDA   #$20        Space
C24B 17   FF41          LBSR  OUTCH
C24E 20   F4            BRA   TEST1

                  *
                  * Output left nibble of A as a hex digit.
                  *    Alters only A Register
                  *

C250 44         OUTHL   LSRA              Shift to right nibble
C251 44                 LSRA
C252 44                 LSRA
C253 44                 LSRA

                  *
                  * Output right nibble of A as a hex digit.
                  *   Alters onla A Register
                  *

C254 84   0F    OUTHR   ANDA  #$0F        Strip left 4 bits
C256 8B   30            ADDA  #$30        Convert to ASCII
C258 81   39            CMPA  #$39        Alpha??
C25A 23   02            BLS   OUTH1
C25C 8B   07            ADDA  #7          Adjust alphas
C25E 16   FF2E  OUTH1   LBRA  OUTCH       Print and return
```


18

'68' Micro Journal

```
                        * Output contents of A as 2 hex digits.
                        *   Alters only A Register
                        *
C261 34  02     OUTHA   PSHS    A       Save A
C263 BD  E8             BSR     OUTHL   Print MS nibble
C265 35  02             PULS    A       Get it again
C267 20  E8             BRA     OUTHR   Print LS nibble
                        END

0 ERROR(S) DETECTED

SYMBOL TABLE:

CAPLOK 011A  CHECK  C0AC  CK1    C0C2  CK2    C0CA  CKCON  C0DC
CKI    C0D2  CKSHFT C0DA  CURSOR 0088  DEL    C23D  DEL14  C23A
DEL20  C237  ESCAPE C000  ESCFLG 03FF  ESCS   C008  GETCOL C0E5
GETKEY C233  INI    C19F  INCH   C18D  INCHNE C193  INKEY1 C188
INKEY2 C18D  INKEY5 C1DE  INKEYB C1CD  INKEYI C186  INKY0  C1AD
INKY01 C1AA  KEYREG 03FD  KEYSCR 0152  LDELAY C0FB  LOOKTB C0FD
NOKEY  C21C  OUTCH  C18F  OUTHI  C25E  OUTHA  C261  OUTHL  C250
OUTHR  C254  PC0    C022  PC1    C05B  PC3    C035  PC4    C047
PIA1   FF00  POLC1  C05F  POLC2  C06B  POLC3  C006  POLC4  C0BD
POLC5  C094  POLC6  C0A8  POLC7  C0AB  POLC8  C0F6  POLCAT C013
RPTCTR 03FE  SPC    C0D6  SPEC   C04B  SPEC1  C053  SS1    C22F
STAT   C1EE  STAT1  C1FB  STAT2  C203  STAT3  C210  STATS2 C221
TEST   C23E  TEST1  C244
```

CC FORTH

James Perotti
163-D Pine Grove Hts.
Athens, Oh. 45701

The Color Computer is finally receiving the recognition which it deserves. A full blown version of Forth has been developed for the 80-C by Charles Eaker, Ph.D; **ccForth**, as it is called, is a special version of Eaker's popular X-Forth which runs on the Flex operating system. Frank Hogg Laboratory sells **ccForth** for $99.00; it is useable with 16K memory. We expect good things from Ph.D's and in this case we get an excellent package. The core of **ccForth** automatically loads from the disk; but also included are a number of additional programs (words). The package includes 3 editors: a screen editor, a line editor and a block editor. A Forth assembler is available on the disk, permitting the user to incorporate machine language subroutines into Forth programs. To demonstrate the usefulness of Forth, Eaker includes a General Ledger program as an example of an extended programming example. There are routines for double precision numbers, a printer routine, a game, and some short programs to create music or sound effects.

Many of us believe that Forth is the fastest way to write game programs. Basic programs execute too slowly; machine language programs are too difficult to write. Forth programs can be written quickly and will execute much more rapidly than Basic, but not as quickly as machine language. The Color Computer version of **ccForth** simplifies programming by providing a one "word" subroutine for the most used graphics modes. Sound effects are also provided. The joysticks can be read at Hex 15A & 15B. So **ccForth** gives the user a good start for creating great games.

Forth ain't easy. Eaker starts slowly but builds rapidly, forcing the learner to retain or review. The documentation is very good; it is clear and thorough. But it is not a good means to learn how to program in Forth; too much is assumed. Leo Brodie's **Starting Forth** (Englewood Cliffs, N.J.: Prentice-Hall, 1981) is an excellent introduction to Forth; it is almost completely compatible with ccForth; reading both books together was a great help. Eaker's manual treats us like intelligent adults; it is not the introductory course to a cut-down version of Forth. Dr. Eaker is teaching us at an advanced level; it is nice to be treated this way, it is nice to see the Color Computer treated with respect. The manual is not perfect; it took some searching and some experimenting to figure to store things on the disk and how to print files. Some of this information is not in the manual but appears on the disk. To access this additional information load the screen editor with 'Decimal 22 LOAD' and then read the first four screens with that editor. I might also mention a problem I had with the editors; as new screens were created, ccForth automatically added my initials and the date to the screen, writing right over my work and causing errors. This problem was easily remedied by always leaving the second lines of screens blank; the initials and the date are then written to the blank line with no problem.

Forth uses Reverse Polish Notation to great advantage. The traditional logic for programming calculators was developed by logicians at the University of Warsaw. Reverse Polish Notation (RPN) is also a polish product, but it is nothing like the traditional approach. We are accustomed to seeing this sequence: number function number equals answer, e.g., 5 x 4 = 20. Reverse Polish Notation goes like this: number1 number22 function answer, e.g., 5 4 x 20. The function follows the numbers on which it operates, executing the calculation. RPN eliminates the need for parentheses and equals signs; in complex problems RPN better follows the flow of the calculation and saves many keystrokes. But when first seen, RPN seems a bit bizarre.

RPN is a stack oriented approach; Hewlett Packard calculators and Forth use stacks to process calculations. Imagine a stack of plates; plates must be placed on top of the stack and removed from the top of the stack. Computers set aside areas of memory which they treat as stacks, handling numbers in the stack on a last-in first-out basis. In RPN the top two numbers on the stack are added, subtracted, multiplied or divided. As an example of the power of this approach, let us look at a simple problem:

$$6(3+2)\ (4x5)\ /\ (5x6) = 600/30 = 20$$

Example of **STACK BASED** Arithmetic

Keys Entered	Top of STACK ------>
6	6
3	3 6
2	2 3 6
+	5 6
x	30
4	4 30
5	5 4 30
x	20 30
x	600
5	5 600
6	6 5 600
x	30 600
/	20 (the Answer)

TABLE 1

Table 1 shows how Forth deals with this problem utilizing RPN. The approach is a trifle backwards, perhaps, but effective. Once you begin to think in terms of applying functions to numbers, the traditional way seems dumb. Hewlett Packard's success with calculators is a measure of the power and acceptance of RPN.

ccForth uses two stacks in many applications. The 6809 is great for Forth; it has two stacks and both are 16 bit. It must be a mess to write Forth for the 6502. In **ccForth** the regular stack is used as described above; the second stack is designated the "return" stack since it often holds the return addresses of words: it also

holds the parameters for loops, and generally keeps numbers or addresses which are in the way. Forth provides the means to change the order of numbers on the stacks and to move numbers from stack to stack. The trick in all of this is to keep track of what's where on which stack. Flowcharting is only for beginners in Basic; worksheets are a necessity in Forth.

Forth builds with "words"; these are subroutines or program modules which are either part of the ccForth core, or are words which you create for your own use. Forth is a language which is loved or hated. Some of us love it because we can combine words into modules of instructions, and then define the modules as words. Forth is therefore extremely compact; programs can be written very quickly because the words are so powerful. Forth is continuously being created; you develop your own language by defining words to suit your particular purposes. ccForth is just the starting point for each of us to begin with; my words will make my Forth different from yours.

To illustrate how Forth gets created, let us pick up on one of the examples in the manual which converts numbers to and from decimal, hex or binary. ccForth has the word BASE to enable us to switch the number system being used. Let us set Forth to base 10 with this definition of the word DECIMAL:

```
: DECIMAL 10 BASE ! ;
```

The colon at the front of the definition and the semicolon at the end are delimiters specifying that a definition is enclosed. DECIMAL is the word being defined. In typical backward RPN fashion, 10 BASE says that DECIMAL is to become equivalent to base 10. The exclamation point is read "store", it indicates that 10 is to be stored in the address pointed to by the variable BASE. To facilitate the use of Hex, the word is defined:

```
: HEX 16 BASE ! ;
```

Easy, huh? Forth uses a dot or a period to display the number on top of the stack, so:

```
DECIMAL 92 HEX .
```

converts decimal 92 to Hex 5C and displays it on the screen. Binary is the same deal:

```
: BINARY 2 BASE ! ;
```

Now for the 'piece de resistance', What if we combine those words making one word which will convert decimal to both Hex and Binary? This demonstrates how modules can be combined to form ever more powerful and more abstract words.

```
: CONVERT DUP HEX . BINARY . ;
```

The word DUP makes another copy of the number at the top of the stack, that is necessary because each dot removes the top number. CONVERT simplifies working with addresses; if you write "32 CONVERT" the old BD-C will display 20(hex) and 100000(binary). Isn't that great? Well there is one little problem with it. When CONVERT is used in this form, it leaves us in base 2. Binary does not react well to two's or three's, much less nine's. We had better redefine CONVERT so that we end up in safe old decimal:

```
: CONVERT DUP HEX . BINARY . DECIMAL ;
```

No big deal. Funny surprises can occur when you think that you're working in decimal and it turns out to be hex; in binary everything but one's and zero's are rejected.

Forth fans love to create their own vocabulary. This permits them to work at a very high level with a personal set of "macro" words. Some people hate Forth for the features which others love. A word which combines other words is difficult to understand because it does a number of things simultaneously. It is the opacity and the complexity of these words which instills hatred. The RPN logic and the reliance on the stacks drive some people up the wall, RPN seems backwards and perverse; Forth even more so. The use of the stacks is regarded as an advantage to Forth fanatics, because it gives Forth speed. The reliance on the stacks is an aggravation to Forth philistines. These people prefer to use variables to keep track of their strings and numbers. While Forth also employs variables, it is considered uncool to use them. Instead of creating a variable, the Forth user pushes the number or address onto the stack. There is a tradeoff between degree of difficulty and speed.

Unquestionably, Forth is difficult to learn and requires technical skill when programming. The payoff comes in speed. Forth is generally believed to be the fastest way to write programs. It is also believed to be the fastest executing language, with the exception of machine language. Whether you love Forth or despise it, ccForth is an excellent adaptation of that language for the Color Computer. For those of you with $99, lots of time and a strong desire to master what might become the language of the computer elite, ccForth will give you the opportunity to rise to greatness.

"C" User Notes

Norm Cosmo
3 Pryor Road
Natick, MA 01760

Last month we talked briefly about the "struct" data type of C. Now let's go over it again with a using a somewhat more formal approach. We will finish off with a program that will show structs in action.

The struct is an aggregate data of the user's own design. It is declared

```
struct tag  {
            .
            .
            data declarations;
            .
            .
            } s_name;
```

This is the most general form, ignoring initializers. The components have the following meanings.

tag
: This is a "tag" name given to the struct since you may have other struct's with different formats. It allows you to use this struct as a template for other declarations.

data declarations
: These are the declarations for the "fields" of the struct. They may be base data types such as char's and int's; or they may be complex data types such as other structs or unions (which we will cover next month).

s_name
: This is the name of an actual variable of the type "struct tag". It causes the necessary storage to be allocated for the variable.

Both tag and s_name are optional components as long as at least one is used. In the above example we just happened to define the struct and declare its storage at the same time. If there were to be no other

references to this particular struct then we could have left out the tag. Similarly, if we had just wanted to make a template that would be referred to by other declarations, then we could have left off s name. To create variables of this type we use the declaration

```
struct tag variable_name;
```

When using the struct in a program, we must refer to both its struct and member names using the syntax

```
struct_name.member_name
```

Here are some examples of the forms struct definitions and declarations can take.

```
/* a boiler plate */
struct date {
          int day, month, year;
          };

/* declaration of a variable */
struct  {
        int day, month, year;
        };

/* a boiler plate and a variable declaration */
struct date {
          int day, month, year;
          } birthday;

/* a declaration using the boiler plate */
struct date birthday;

/* a pointer to a structure */
struct date *birthday;

/* structs can be static too! */
static struct date birthday;
```

The examples could go on and on, but these give the flavor of what can be done. Structs may also be globals just like any other data type.

There are really two formats for referring to structs when you use them in a program. If you are working with an actual struct variable the syntax, which was given earlier, is

```
struct_name.member_name
```

if, on the other hand, you are working with a struct indirectly, via a pointer, then the syntax is

```
pointer_to_struct->member_name
```

You will probably find that this is the syntax that will get the heaviest use. In all the code I've peeked at, the majority of struct references seem to be through pointers.

The program that will serve as a vehicle for learning about structs, simple binary trees and recursion is one that finds all the unique words in a file and keep a count of their occurrence. To top it all off, it may even be useful! The program was written using the INTROL compiler.

Since we don't know how many words there will be ahead of time, or how big the biggest word will be, we will use dynamic data structures. We will blindly (but smartly) use the binary tree. The definition of a "node" in the tree will be

```
struct node {
          char *word;
          int wrdcount;
          struct node *left, *right;
          };
```

Note the self reference. We haven't even finished the definition of struct node when we declare two pointers to it. This is legal since the size of a pointer is independent of the object it points to. It will always be two bytes for a 6809.

Here is a quick definition of how the program will work.

1) The program will request a file name from the user an open it up.

2) It will then read the file one line at a time.

3) Each line will be parsed for an alpha sequence, which we will call a word.

4) If a word exists within the tree, it means that it has been encountered previously and the count for that word is incremented.

5) If the word is not encountered in the tree, then storage is allocated for both the word and a new node and the node is inserted into the tree with the count set to one.

5) When the input file is empty, the program will print out all the "words" it found and the number of times that they occured in the file.

A limitation of this program will be that a word will be defined as a sequence of alpha characters bounded by ANY nonalpha characters. Case of the alpha characters will be ignored.

The complete program is contain at the end of the column. Now let's look at what it does by studying the major functions.

main()

Prompts for a file and tries to open it, failure is reported and the program quits.

The open file is read a line at a time using fgets(). The line is parsed for tokens (words) by getoken(). The variable next is used as a place holder within the line. It is updated by getoken().

Each token is then passed to insert() which traverses the tree trying to find a match.

When the file is finally empty the tree is printed out alphabetically by print_tree() and the program is finished.

getoken()

Parses the line for a "word". It skips over any leading non alpha characters until it finds an alpha or the end of the string. If the end of the string is found it indicates so by returning NULL.

The word is then transferred to tok, with each character converted to lower case. The word ends when the next non alpha or the end of string is found. In either case, getoken() terminates the string in tok and returns the pointer to the termination character.

insert()

Builds an "inorder" binary tree recursively. The way the function is declared,

```
struct node *insert(word,tnode)
```

tells the compiler that insert() returns a pointer to a struct of type node.

When insert() is called, it is passed a word to look up or place in the tree and a pointer to a struct node.

If that pointer is NULL then it means one of two things. Either the tree is empty or insert() has finally traversed down to the end of one of the branches. In either case a new node must be created.

Storage is created dynamically from the heap by the function alloc(). Alloc() is passed the size of the desired storage in bytes. If the storage can be allocated successfully alloc() returns a pointer to it. Otherwise it returns NULL. In that case, insert() calls all err() which flags the problem and aborts the program.

If the pointer is not NULL, word is compared to the string pointed to by tnode->word. If they are equal, tnode->wrdcnt is incremented. If word is less than tnode->word, then insert() branches to the left by calling itself with the parameters word and tnode->left. Otherwise it branches right.

The result is an "inorder" tree which has the characteristic that, at any node in the tree, all the words to the left of that node will be lexically smaller than the word contained in the node, and all words to the right will be larger.

Be aware that sizeof() isn't a real function. It's a compiler directive. The compiler will calculate the size of sizeof()'s argument which is usually a complex data object, and then replace sizeof() with a constant equal to the calculated value.

Also note that when dynamically allocating array space for strings, you must add 1 to the size returned by strlen() since it ignores the NULL terminator, which you must allow for when saving the string.

print tree()
Recursively prints out the tree in order. It is passed root, which points to the "root" of the tree. Note that the root of the tree is not the first element of the tree in a lexical sense. rather it was the first word that insert tried to find.

If the pointer passed to print tree() is not NULL, then print tree traverses to the left in same manner that insert() did. This is repeated until the left most node is found, which will contain the smallest word, lexically, in the whole tree.

At this point, print tree() prints out the word and its tally. It then traverses to the right of the node.

This program points out rather vividly the utility of dynamic data structures and the ease with which they can be handled with structs. I think it lends some credence my earlier opinion that the pointer syntax is typically used more often that the syntax for an actual structure variable.

I am not going to pretend that I can make recursion absolutely clear to everyone, since I am rather convinced it is one of those "lighting" concepts. We are all familiar with those. We study them to no avail. Then while we are doing something totally unrelated the understanding pops into our mind as suddenly as a flash of lightning. The only thing I can say is to study insert() and print tree() being aware that each time they call themselves a new stack frame is created. Hence, in insert() for example, the variable word is unique for each call.

As an interesting side comment. If you want to print out the tree in reverse order, then merely interchange print_tree()'s two calls to itself.

WHAT'S NEW

I recently received Introl's OS-9 compiler. It is a very nice package as delivered. I will be trying it out shortly. I have seen it run on a friend's machine and all went smoothly. The OS-9 interface is especially nice. There are C functions for forking off a child process, sleeping, sending signals to another process, intercepting a single and quite a few others. If you want to experience a really dynamite programming environment then try C and OS-9.

As with the FLEX version, you get the compiler, an assembler, a linker, a librarian, the standard library (in binary) and all the C or assembler sources for the functions contained in the library. This is a real bargain for the price.

The other night I had a nice long chat with the folks at Word's Worth. They have not been idle. Release two is now being shipped. It has a number of improvements and additions over their initial release. Here is the list.

the logical || and && operators

the data type short, which is 8 bits long

the complex assignment operators
-= += |= /= %= &= ^=

decimal, octal and hexidecimal literals

printf now supports \n \r \t \f \b \ddd (octal)

#ifdef, #ifndef and #endif

the functions alloc(), free(), fgets() and fputs()

The number of code sequences that the peep-hole optimizer now recognizes and replaces has gone from 5 to 20. According to Word's Worth the improvement in code density is not much, but the speed improvement is signicant. The Sieve program now runs in about 40 seconds on a 1Mhz machine.

Release 2 will be followed by release 2.1 which will include

the data types long and unsigned

the ?: assignment operator

#define variable substitution

#if and #else

Best of all, the update cost from release 2 to release 2.1 will be very minimal, just the media cost.

They are also working on a relocating macro assembler. This should be available during the first quarter of 1983. The assembler, which will include a linker, will be integrated into the C compiler package. It will also be sold separately in the $75 dollar price range. The compiler will be optimized to work with the assembler.

Word's Worth would also like to start a public domain C library. The idea is that people would submit programs that would be grouped together and sold in distribution "kits". The cost of the kit would be somewhat higher than the cost of the median, in order to provide a small royalty to the people who submitted the programs. Sounds to me like an idea that should be given some serious thought and support.

WHAT WENT WRONG

Profound apologies for the program listing in the last column. Apparently some of the characters very common in C programs had an absolute field day wrecking havoc on the printer!. I don't want to reprint the listing right now so let me tell you what to look out for.

First of every "<cr>#define" got rendered as "20efine". This made mess of the defines. The OR symbol got rendered as a backwards P. That really didn't do too much damage. The really biggy was the deletion of the back slash character. This can make the case comparison characters appear somewhat cryptic to say the least.

The listing will be published again at a later date. I have also spoken to Don and we think that we have a system that will prevent this from happening again.

THE FUTURE

A language column like C NOTES can have a very short life time compared to something like Ron's FLEX column. Let's face it, the subject matter is much more limited. After explaining the syntax of the language just about all that is left are hints, programming examples and news about what the vendors are up to.

I need your help in the form of suggestions. I really don't mind doing the work, but I could use some feedback as to what you would like to see. I have received only four or five letters since the column began.

I have some ideas. One is to let you readers use the column as an information exchange of sorts. If you have run across a bug and have found a way around it, then write it up and share it. Credit will always be given to you. Got a hint to make life easier? Share it, we are all lazy! Again, let me stress that you only have to supply the idea. Let me try to flesh it out, you'll still get the credit.

I recently started back to night school and am developing a desire to write some OS-9 articles. This will cause the column to go bimonthly more often that I would prefer. I have tried to make the columns substantial and feel that a decent column bimonthly is better than fluff monthly.

Next time we will tie up a few loose ends and that will finish up most of what I had planned to "teach" about the language. What will be left unsaid can be learned by the more advanced programmer quickly anyway. See you then.

The Word Tree Listing

```
/*
 * A program to find the unique words in a
 * file and keep a count of their occurances.
 *
 * Notes    case is ignored
 *          a "word" is one or more alpha
 *          characters seperated by any
 *          nonalpha characters.
 */

#include "STDIO.H"
#define LSIZE 256

/*
 * this is the binary tree's node
 * structure definition
 */
struct node {
            char *word;
            int wrdcnt;
            struct node *left, *right;
            };


main()
    {
    char line[LSIZE], token[LSIZE], *next;
    FILE *file_id;
    struct node *root;

    /* plant the tree */
    root = NULL;

    /* get the file set up */
    printf("Enter the file name: ");
    gets(line,LSIZE);
    if ((file_id = fopen(line,"r")) == NULL)
        {
        printf("\nERROR opening file %s\n",line);
        exit();
        }

    /* now build the word tree */
    while (fgets(line,LSIZE,file_id) != NULL)
        {
        /* set the line scanner to start-of-line */
        next = line;

        /* now scan the line */
        while ((next = getoken(token,next)) != NULL)
            root = insert(token,root);
        }

    /* and finally, print out the list of words */
    print_tree(root);
    }


/*
 * print out the tree recursively
 */
print_tree(tnode)
    struct node *tnode;
    {
    if (tnode != NULL)
```

```
        {
        print_tree(tnode->left);
        printf("%s %d\n",tnode->word,tnode->wrdcnt);
        print_tree(tnode->right);
        }
    }


/*
 * get the next token by:
 *
 * 1) skipping over any leading nonalpha's
 * 2) changing all alphas to lower case
 * 3) returning NULL if end-of-string,
 *    otherwise a pointer to the next
 *    nonalpha
 */
char *getoken(tok,strng)
    char *tok, *strng;
    {
    /* skip over any leading nonalphas */
    while (!isalpha(*strng))
        if (*strng++ == NULL)
            return(NULL);

    /* get the token */
    while (isalpha(*strng))
        if (*strng == NULL)
            break;
        else
            *tok++ = tolower(*strng++);

    /* terminate the token and return */
    *tok = '\0';
    return(strng);
    }


/*
 * search the tree for a word, if found
 * then bump the count, otherwise insert
 * the word into the tree
 */
struct node *insert(word,tnode)
    char *word;
    struct node *tnode;
    {
    int cnd;

    if (tnode == NULL)
        {
        /* create a new node */
        if ((tnode = alloc(sizeof(struct node))) == NULL)
            all_err();

        /* create an array for the word */
        if ((tnode->word = alloc(strlen(word)+1)) == NULL)
            all_err();

        /* save the word and init the new node */
        strcpy(tnode->word,word);
        tnode->left = tnode->right = NULL;
        tnode->wrdcnt = 1;
        }
    else
        {
        /* test the new word agains the current node */
        if ((cnd = strcmp(word,tnode->word)) == 0)
            tnode->wrdcnt++;
        else if (cnd < 0)
                tnode->left = insert(word,tnode->left);
             else
                tnode->right = insert(word,tnode->right);
        }
    return(tnode);
    }


/*
 * flag the allocation error
 * and quit
 */
all_err()
    {
    printf("\n\007ERROR on allocation\n");
    exit();
    }
```

Winchester

ADDING A WINCHESTER (CONCLUSION)
David J. Graves
311A Towne Building/D3
University of Pennsylvania
Philadelphia, Pa. 19104
(215) 898-7951

With the hardware portion of the Winchester installation completed, all that remains is the writing of appropriate software. Although I mentioned his name in the previous installment, I should again give credit here to Robert Zeff, who supplied me listings of his routines early this summer. I have re-written them from the viewpoint of someone who already has a floppy disk based SS-50 system and wishes to add a co-resident Winchester as easily as possible rather than boot-up a solo hard disk more or less from scratch. I will make my software available on disk for $17.50, and readers may choose between the original Zeff version and mine depending on which suits their own systems better. Our hardware interfaces also differ significantly, with mine being designed to plug into an existing system and his being more suitable for building a new and relatively inexpensive system from scratch.

The first task that the software must do is to format the disk. At this point, I will refer readers who are unfamiliar with the concept of sector interleaving to Leo Taylor's article in the April issue of this journal. Basically, the concept involves putting the sectors on a disk in non-numerical order to speed up the reading and writing process. The first program given here (WINPMT) does just that, and in addition provides a convenient method for determining which interleave factor results in the fastest disk reads or writes. When first starting up, I suggest that the prompts for number of tracks, sectors per track, and interleave factor be answered by 256, 32, and 25 respectively. Later, when the system is up and running, one side of the disk can be used to experiment with different interleave factors, and in this case I suggest that a small number of tracks (e.g. 10) be specified. This permits one to format a portion of that side of the disk quickly instead of waiting the several minutes required to format all 256 Tracks. Relative read and write speeds can then be tested on this abbreviated disk surface at some experimental interleave factor.

I copied TSC's Extended Basic (79 sectors) from side 0 (which is drive 2) to side 1 (drive 3) of the Winchester with various interleave factors formatted on drive 3. Then I loaded this fairly lengthy program from drive 3. The results were interesting. Program loading times increased from slightly over 5 to 6.2 seconds as the interleave was increased from 1 to 23. There was then a precipitous drop to about 4.9 seconds at an interleave of 24 and a further slight decrease to 4.7 seconds at 25. Other interleaves up to 32 produced 5.1 or 5.2 second execution times. In contrast, the time to COPY,2,3,X@ASIC was fastest at an interleave of 1 (12.4 sec.) and relatively slow (14.4 sec.) at an interleave of 30 with a gradual increase in time between the two values. On the other hand, copying a file from drive 3 to drive 3 under another name was almost the same speed (12.9 vs. 12.6 sec.) at interleaves of 2 and 25. I guess

the moral is that different interleaves are needed to optimize different processes. Since disk reads are undoubtedly the most common process, I interleave at 25. Incidentally, the copy time for XBASIC from one minifloppy to another was 63.4 sec., and 14.6 sec. was needed to load it. The Winchester is therefore about 3 or 4 times faster than a floppy even if not optimized by careful interleaving and despite Leo's comments about the slowness and non-I/O bound character of FLEX.

Once the proper interleave has been found and both drives of the Winchester have been formatted with their full 256 tracks, WINFMT.CMD will not be needed again unless one wishes to quickly erase a whole side and start over. WINFMT is somewhat of a Winchester version of NEWDISK, but it does not write a bootstrap loader on the disk as NEWDISK does for floppies. It will also not map out defective sectors, although I have not yet had any need to do this. I will put in a plug for TSC's "Diagnostics" disk, which contains a utility for doing this as well as a utility called FILETEST, which checks for bad spots, EXAMINE, which I have used many times to look at and change bytes on a disk, (hard or floppy) and a number of other useful utilities as well.

PATCHING FLEX

A second program is all that is needed to begin using the hard disk. I have taken a rather simple expedient to permit both hard disk and floppies to be simultaneously accessible. There is a jump table at $DE00 in FLEX which vectors to seperate routines for reading, writing, verifying, etc. the sectors on a disk. (See p. 6 of the FLEX general notes). I have substituted jumps to my routines in the Winchester driver program, which I call HRDDRV. There the File Control Block (FCB) is examined to see if drives 0 or 1 (floppies) or drives 2 or 3 (the Winchester) are being addressed. The routines then either branch back into the old floppy drivers or into the new Winchester drivers. All of the program HRDDRV fits in less than 2K of memory. I use one of the two memory chips added as part of the interface described last month (from $E800-$EFFF) for this program. In this location it does not interfere with any other programs or use up any of the normal 56K of memory. The second 1.75K memory chip is not needed, but it is convenient to store the Winchester bootstrap in EPROM, for example. Since the bootstrap and loader programs are not needed if one is content to boot the floppies and then load the Winchester drivers, I do not give them here. If there is enough interest I will write an additional article and I will put them on the disk along with the other two.

Adding the drivers to FLEX is quite simple. I have renamed HRDDRV in its assembled version HARD.CMD. When the floppy version of FLEX is up and running, one simply types HARD. The drivers load and overlay the eight disk entry points in FLEX and control is then returned to FLEX. A CAT,2 or CAT,3 command will now show 8160 free sectors for each drive. Since I usually want to make drive 2 of the Winchester the working and system drive, I have a second file called HARD.TXT. It is created as follows:

```
+++BUILD,HARD.TXT
=HARD                    (note that this means HARD.CMD)
=ASN,W=2,S=2
=#
```

Now it is a simple matter to load the drivers and simultaneously assign drive 2 as both working and system drive by typing EXEC,HARD. An advantage of this overlay procedure is that a system with the original version of FLEX and floppies alone (even 4 floppies if desired) can be operated, and the Winchester added or substituted for two of the drives whenever it is needed. The noise and heat production of the Winchester may sometimes make this option desirable.

Adding this Winchester to my system has dramatically improved its capabilities and eased my work. At present, I use only drive 2 to store programs and devote drive 3 to backing them up. This has the advantage that the file fragmentation which takes place as files are deleted and added can be eliminated by copying everything to drive 3, erasing (or reformatting) drive 2, and then re-copying everything back to drive 2. It would be most appropriate when first putting files on the Winchester to put those which will always be kept (languages, utilities, etc.) on first, followed by those files which may well be deleted or edited. This should help to minimize file fragmentation.

I hope that this discussion will encourage others to go the Winchester route. I would like to hear the experiences of those who try it. Part of my motivation is selfish. I can already see the need for other utilities for carrying on partial disk copying, efficiently examining parts of the disk catalog, etc., and hope to entice someone to help with the job. Writing drivers to configure seperate Winchesters as drives 2 and 3 also would be a commendable effort. One of the first jobs should be to alter NEWDISK so that someone could not unknowingly or accidentally apply it to the Winchester. Although I have not tried this, I expect that it could be disasterous. Who would like to volunteer in the software effort ?

```
* HARD DISK FORMAT ROUTINE.  WRITTEN BY R. ZEFF 5/26/82 AND         *
* MODIFIED BY D.J. GRAVES 7/12/82 WITH SELF-CONTAINED DRIVERS        *
* AND ADDED COMMENTS.  VALID FOR WESTERN DIGITAL WD1000 CON-         *
* TROLLER AND FLEX9.  PROGRAM FORMATS, LINKS, REPORTS ERRORS, &      *
* SETS UP DIRECTORY AND SYSTEM INFO SECTORS.  IT DOES NOT MAP        *
* OUT BAD SECTORS.  USER CAN CHOOSE LESS THAN 32 SECTORS/TRACK       *
* AND 256 TRACKS FOR EXPERIMENTATION.  DETERMINE INTERLEAVE          *
* FACTOR BY TRIAL AND ERROR FOR FASTEST RESPONSE.                    *
* DAVID J. GRAVES (215) 898-7951                                     *
* 311A TOWNE BUILDING/D3                                             *
* UNIV. OF PENNA.                                                    *
* PHILADELPHIA, PA. 19104                                            *
```

Portions Copyrighted 1982 Robert Zeff

```
CD03 WARMS   EQU  $CD03   THESE ARE ALL STANDARD FLEX EQUATES
CD39 OUTDEC  EQU  $CD39
CD15 GETCHR  EQU  $CD15
CD1B INBUFF  EQU  $CD1B
CD33 SETEXT  EQU  $CD33
D406 FMS     EQU  $D406
CD2D GETFIL  EQU  $CD2D
CD18 PUTCHR  EQU  $CD18
CD0F OUTCH   EQU  $CD0F
CD1E PSTRNG  EQU  $CD1E
CD24 PCRLF   EQU  $CD24
CD48 INDEC   EQU  $CD48
CD42 GETHEX  EQU  $CD42
CD45 OUTADR  EQU  $CD45
CD4E STAT    EQU  $CD4E
CD3F RPTERR  EQU  $CD3F
CD3C OUTHEX  EQU  $CD3C
CD27 MATCH   EQU  $CD27
CC0E DATE    EQU  $CC0E

E8C0 HBASE   EQU  $E8C0   SYSTEM DEPENDENT: FIRST ADDRESS OF DISK PORT
E8C0 RDATA   EQU  HBASE   REGISTER 1 FOR READ PURPOSES
E8C8 WDATA   EQU  HBASE+8 REGISTER 1 FOR WRITE PURPOSES
E8C1 ERROR   EQU  HBASE+1 REGISTER 2, ETC. FOLLOW IN ORDER
E8C9 WPRE    EQU  HBASE+9 SEE WD1000 MANUAL FOR DEFINITION OF PURPOSES
E8C2 RSCNT   EQU  HBASE+2
E8CA WSCNT   EQU  HBASE+10
E8C3 RSNUMB  EQU  HBASE+3
E8CB WSNUMB  EQU  HBASE+11
E8C4 RCYLL   EQU  HBASE+4
E8CC WCYLL   EQU  HBASE+12
E8C5 RCYLH   EQU  HBASE+5
E8CD WCYLH   EQU  HBASE+13
E8C6 RSDH    EQU  HBASE+6
E8CE WSDH    EQU  HBASE+14
E8C7 STATUS  EQU  HBASE+7
E8CF COMAND  EQU  HBASE+15
          0020 SECTRK  EQU  32
          00FF TRKMAX  EQU  255
          00AF FORMAT  EQU  %10101111 NOTE THAT THIS AND ALL OTHER COMMANDS MUST
          FFDF PRECMP  EQU  !32       BE GIVEN AS THEIR COMPLEMENT SINCE DATA IS
          00E0 RESTSL  EQU  $E0       NOT RE-INVERTED BY THE 74LS245 DRIVER
          00EF REST    EQU  $EF

0000                      ORG  0

0000 20   01      START   BRA  SD
0002 02                   FCB  2        VERSION 2

                  * INITIALIZE THE DRIVE

0003 86   FF      SD      LDA  #$FF
0005 B7   E8CE            STA  WSDH     CHOOSE DRIVE-SIDE, HERE AS ZERO
0008 86   DF              LDA  #PRECMP
000A B7   E8C9            STA  WPRE     SET PRECOMPENSATION TRACK
000D 86   FF              LDA  #-1
000F B7   E8CD            STA  WCYLH    CYLINDER HIGH IS ALWAYS ZERO ON 5 MEG DRIVE
0012 86   E0      REST1   LDA  #RESTSL
0014 B7   E8CF            STA  COMAND   DO A SLOW RESTORE
0017 7D   E8C7    REST3   TST  STATUS
001A 2A   FB              BPL  REST3    CHECK BUSY FLAG
001C 86   EF              LDA  #REST
001E B7   E8CF            STA  COMAND   NOW DO DUMMY FAST RESTORE TO RESET SEEK SPEED
0021 7D   E8C7    REST2   TST  STATUS
0024 2A   FB              BPL  REST2    BUSY?
0026 C6   00              LDB  #0

                  * MAIN FORMATTING PROGRAM BEGINS HERE

0028 8E   03BE            LDX  #RDYMES  READY TO FORMAT? Y/N
002B BD   CD1E            JSR  PSTRNG
002E BD   CD15            JSR  GETCHR
0031 81   59              CMPA #'Y
0033 1026 CD03            LBNE WARMS
0037 8E   0362    WDRV    LDX  #DRVMES  WHICH DRIVE? 2/3
003A BD   CD1E            JSR  PSTRNG
003D BD   CD15            JSR  GETCHR
0040 80   30              SUBA #'0
0042 7F   02D8            CLR  SIDEHD
0045 B7   02D9            STA  SIDEHD+1
0048 81   03              CMPA #3       SEE IF VALID DRIVE NO.
004A 22   EB              BHI  WDRV
004C 81   01              CMPA #1
004E 23   E7              BLS  WDRV
0050 34   02              PSHS A
0052 30   7D              LEAX -3,S
0054 BD   046A            JSR  SELECT   SELECT THE DRIVE
0057 32   7F              LEAS -1,S     AND TWEEK NUMBER
0059 8E   03DC            LDX  #TRKMES  HOW MANY TRACKS? (1 TO 256 PERMITTED)
005C BD   CD1E            JSR  PSTRNG
005F BD   CD1B            JSR  INBUFF   GET DECIMAL NUMBER
0062 BD   CD48            JSR  INDEC    AND CONVERT TO BINARY
0065 30   1F              LEAX -1,X     (TRACKS ACTUALLY RUN 0 TO 255)
0067 BF   02DB            STX  TRKCNT
006A 8E   03F7            LDX  #SECMES  HOW MANY SECTORS? (1-32)
006D BD   CD1E            JSR  PSTRNG
0070 BD   CD1B            JSR  INBUFF   ETC. AS ABOVE
0073 BD   CD48            JSR  INDEC
0076 1F   10              TFR  X,D      PUT SECTOR NUMBER IN D
0078 F7   02DA            STB  SECCNT   AND SAVE L.S.BYTE
007B BD   03E1            JSR  CALVE    CALCULATE INTERLEAVE TABLE
007E 4F                   CLRA          START WITH TRACK ZERO
```

```
007F 34    02             FILL     PSHS  A
0081 BD    01C2                    JSR   TFORMT      FORMAT A TRACK
0084 35    02                      PULS  A
0086 B1    02DC                    CMPA  TRKCNT+1
0089 27    03                      BEQ   DOLINK
008B 4C                            INCA
008C 20    F1                      BRA   FILL
008E 7F    02E0           DOLINK   CLR   EOTFLG      CLEAR END OF TRACK FLAG
0091 B6    02DC                    LDA   TRKCNT+1    GET TRACK
0094 F6    02DA                    LDB   SECCNT
0097 FD    02DD                    STD   UNKEND      SAVE END OF SECTOR CHAIN
009A 8E    02E3                    LDX   #WORK+2     CLEAR THE BUFFER
009D 6F    80             DOLIN1   CLR   ,X+
009F 8C    03E1                    CMPX  #WORK+256
00A2 26    F9                      BNE   DOLIN1
00A4 7F    0237                    CLR   DTEMP       START AT TRACK 0
00A7 4F                            CLRA
00A8 7F    0236           DOLIN2   CLR   SECTOR      FIRST SECTOR
00AB 7C    0236           DOLIN3   INC   SECTOR
00AE F6    0236                    LDB   SECTOR      GET SECTOR
00B1 F7    0238                    STB   DTEMP+1     SAVE SECTOR
00B4 B6    0237                    LDA   DTEMP       GET TRACK
00B7 10B3  02DD                    CMPD  UNKEND      SEE IF LAST ONE
00BB 26    08                      BNE   DOLIN4
00BD 73    02E0                    COM   EOTFLG      ALSO END OF TRACK
00C0 CC    0000                    LDD   #0          END OF CHAIN
00C3 20    0B                      BRA   LINKS3
00C5 F1    02DA           DOLIN4   CMPB  SECCNT      SEE IF LINK SHOULD BE TO NEXT TRACK
00C8 26    05                      BNE   LNKS1
00CA 73    02E0                    COM   EOTFLG
00CD 5F                            CLRB              IT IS; LINK TO NEXT TRACK
00CE 4C                            INCA              INCREMENT LINK (NOT TRACK)
00CF 5C                   LINKS1   INCB
00D0 8E    02E1           LINKS3   LDX   #WORK
00D3 ED    84                      STD   ,X
00D5 FC    0237                    LDD   DTEMP
00D8 BD    07C2                    JSR   WRITEC      WRITE IT OUT; IGNORE ERRORS
00DB 27    11                      BEQ   LINK9
00DD 8E    0240                    LDX   #BADMES     OUTPUT AN ERROR MESSAGE
00E0 BD    CD1E                    JSR   PSTRNG
00E3 B6    0237                    LDA   DTEMP
00E6 BD    021F                    JSR   HEXOUT
00E9 B6    0238                    LDA   DTEMP+1
00EC BD    021F                    JSR   HEXOUT
00EF 7D    02E0           LINK9    TST   EOTFLG      END OF TRACK?
00F2 27    B7                      BEQ   DOLIN3
00F4 7F    02E0                    CLR   EOTFLG
00F7 F6    0237                    LDB   DTEMP       SEE IF LAST TRACK
00FA F1    02DC                    CMPB  TRKCNT+1
00FD 27    05                      BEQ   SIRSEC
00FF 7C    0237                    INC   DTEMP       INCREMENT TRACK
0102 20    A4                      BRA   DOLIN2

                          * SET UP THE S.I.R.

0104 8E    02E1           SIRSEC   LDX   #WORK
0107 6F    80             SIF      CLR   ,X+
0109 8C    02F1                    CMPX  #WORK+16
010C 26    F9                      BNE   SIF
010E 108E  0257                    LDY   #DISNAM     WRITE OUT "WINCHESTER" AS DISK NAME
0112 A6    A0             NORNAM   LDA   ,Y+
0114 A7    80                      STA   ,X+
0116 8C    02FC                    CMPX  #WORK+27
0119 26    F7                      BNE   NORNAM
011B FC    02D8                    LDD   SIDEND      PUT ON THE DISK SIDE NUMBER
011E ED    81                      STD   X++         (AS VOLUME NUMBER)
0120 CC    0101                    LDD   #$0101      FREE CHAIN STARTS HERE (PREVIOUS SECTORS = DIRECTORY)
0123 ED    81                      STD   X++
0125 FC    02DD                    LDD   UNKEND      END OF FREE CHAIN
0128 ED    81                      STD   X++
012A B6    02DC                    LDA   TRKCNT+1
012D F6    02DA                    LDB   SECCNT
0130 3D                            MUL               FREE CHAIN SIZE
0131 ED    81                      STD   X++
0133 FC    CC0E                    LDD   DATE        MONTH, DAY
0136 ED    81                      STD   X++
0138 B6    CC10                    LDA   DATE+2      YEAR
013B A7    80                      STA   X+
013D FC    02DD                    LDD   UNKEND      TRACK AND SECTOR MAXIMUM
0140 ED    81                      STD   X++
0142 6F    80             CLRSST   CLR   X+
0144 8C    03E1                    CMPX  #WORK+256
0147 26    F9                      BNE   CLRSST
0149 CC    0003                    LDD   #$0003      POINT TO TRACK 0, SECTOR 3
014C 8E    02E1                    LDX   #WORK
014F 34    06                      PSHS  D
0151 BD    07C2                    JSR   WRITEC      WRITE OUT THE SECTOR
0154 35    06                      PULS  D
0156 8E    02E1                    LDX   #WORK
0159 BD    040B                    JSR   READC       AND CHECK IT
015C 26    22                      BNE   SIFERR
015E 8E    02E1                    LDX   #WORK       FIX LINK FOR END OF DIR. FREE CHAIN
0161 6F    80             SIF6     CLR   X+
0163 8C    03E1                    CMPX  #WORK+256
0166 26    F9                      BNE   SIF6
0168 8E    02E1                    LDX   #WORK
016B 4F                            CLRA
016C F6    02DA                    LDB   SECCNT      LAST SECTOR OF DIRECTORY
016F 34    06                      PSHS  D
0171 BD    07C2                    JSR   WRITEC
0174 35    06                      PULS  D
0176 8E    02E1                    LDX   #WORK
0179 BD    040B                    JSR   READC
017C 1027  CB83                    LBEQ  WARMS
0180 8E    0189           SIFERR   LDX   #SIFMES
0183 BD    CD1E                    JSR   PSTRNG
0186 7E    CD03                    JMP   WARMS

0189 0A0D                 SIFMES   FDB   $0A0D
018B 57 52 49 54                   FCC   'WRITE ERROR ON SYSTEM INFORMATION SECTOR OR DIRECTORY!',4

                          * HERE WE DO THE FORMATTING
                          * 256 BYTES/SECTOR, MAX. 32 SECTORS/TRACK

01C2 7D    E017           TFORMT   TST   STATUS
01C5 2A    FB                      BPL   TFORMT
01C7 F6    02DA                    LDB   SECCNT      SECTOR COUNT MUST BE RELOADED EACH TIME
01CA 53                            COMB
01CB F7    E01A                    STB   WSCNT
01CE 43                            COMA              OUTPUT THE TRACK
01CF B7    E01C                    STA   WCALL
01D2 C6    AF                      LDB   #FORMAT     SEND "FORMAT TRACK" COMMAND
01D4 F7    E01F                    STB   COMAND
01D7 B6    E017           FORMB    LDA   STATUS
01DA 85    10                      BITA  #$10        SEEK COMPLETE?
01DC 26    F9                      BNE   FORMB
01DE 8E    04F5                    LDX   #SECTAB     POINT TO SKEWTABLE
01E1 5F                            CLRB
01E2 B6    E017           FFORMA   LDA   STATUS
01E5 85    08                      BITA  #8
01E7 26    F9                      BNE   FFORMA
01E9 86    FF             FORMS    LDA   #-1         NO BAD BLOCKS MARKED ON DISK
01EB B7    E01B                    STA   WDATA
01EE A6    80                      LDA   ,X+
01F0 43                            COMA
01F1 B7    E01B                    STA   WDATA       WRITE THE SECTOR NUMBER
01F4 CB    02                      ADDB  #2          NOTE THAT 256 BYTES MUST BE TRANSMITTED EVEN THOUGH
01F6 26    EA                      BNE   FFORMA      ONLY 2X # OF SECTORS IS REAL DATA (REST IS GARBAGE)
01F8 7D    E017           FORMA    TST   STATUS      WAIT 'TILL DONE
01FB 2A    FB                      BPL   FORMA
01FD 39                            RTS

01FE 8E    020A           FERR     LDX   #FMESS
0201 BD    CD1E                    JSR   PSTRNG
0204 B6    E031                    LDA   ERROR
0207 7E    CD18                    JMP   PUTCHR

020A 46 4F 52 4D          FMESS    FCC   'FORMAT TRACK ERROR: ',4

021F 34    02             HEXOUT   PSHS  A           SAVE THE CHARACTER
0221 44                            LSRA
0222 44                            LSRA
0223 44                            LSRA
0224 44                            LSRA
0225 8D    04                      BSR   OUT2
0227 35    02                      PULS  A
0229 84    0F                      ANDA  #$F
022B 81    09             OUT2     CMPA  #9
022D 23    02                      BLS   OUT1
022F 8B    07                      ADDA  #7
0231 8B    30             OUT1     ADDA  #$30
0233 7E    CD0F                    JMP   OUTCH

0236                      SECTOR   RMB   1
0237                      DTEMP    RMB   2
0239                      FLAG     RMB   1
023A                      CHNSTR   RMB   2
023C                      CHNEND   RMB   2
023E                      CHNCNT   RMB   2
0240 42 41 44 20          BADMES   FCC   'BAD TRACK, SECTOR AT: ',4
0257 57 49 4E 43          DISNAM   FCC   'WINCHESTER '
0262 57 48 49 43          DRVMES   FCC   'WHICH DRIVE? (2 OR 3) ',4
0279 46 4F 52 4D          DMESS    FCC   'FORMAT DRIVE ',4
0287 0A0D                 SECMES   FDB   $0A0D
0289 48 4F 57 20                   FCC   'HOW MANY SECTORS? (1-32) ',4
02A5 0A0D                 TRKMES   FDB   $0A0D
02A7 48 4F 57 20                   FCC   'HOW MANY TRACKS? (1-256) ',4
02BE 0A0D                 CMESS    FDB   $0A0D
02C0 52 45 41 44                   FCC   'READY TO FORMAT? (Y/N) ',4
02D8                      SIDEND   RMB   2
02DA                      SECCNT   RMB   1
02DB                      TRKCNT   RMB   2
02DD                      UNKEND   RMB   2
02DF                      SIDE     RMB   1
02E0                      EOTFLG   RMB   1
02E1                      WORK     RMB   256

                          * CALCULATE INTERLEAVE TABLE

03E1 F6    02DA           INLVE    LDB   SECCNT
03E4 4F                            CLRA
03E5 C3    04F5                    ADDD  #SECTAB
03E8 FD    0468                    STD   TBLEND
03EB 8E    0449                    LDX   #INTMES     GET INTERLEAVE FACTOR
03EE BD    CD1E                    JSR   PSTRNG
03F1 BD    CD1B                    JSR   INBUFF
03F4 BD    CD48                    JSR   INDEC
03F7 1F    10                      TFR   X,D
03F9 FD    0466                    STD   INTFCT
03FC 86    FF                      LDA   #-1         INITIALIZE ALL SECTORS AS UNUSED
03FE 8E    04F5                    LDX   #SECTAB
0401 A7    80             SETFLG   STA   ,X+
```

```
0403 8C    0468             CMPX   TRKEND
0406 26    F9               BNE    SETFLG
0408 7F    0465             CLR    COUNTR

040B 7F    0467             CLR    RES
040E 8E    06F5             LDX    #SECTAB
0411 B6    0467   SETUP     LDA    RES
0414 B1    E00A             CMPA   SECCNT
0417 25    06               BLO    SETUP1
0419 B0    E00A             SUBA   SECCNT
041C B7    0467             STA    RES
041F F6    0467   SETUP1    LDB    RES
0422 A6    85               LDA    B,X
0424 81    FF               CMPA   #-1
0426 26    07               BNE    B.SE
0428 B6    0465             LDA    COUNTR
042B A7    85               STA    B,X
042D 20    05               BRA    OTHER
042F 7C    0467   B.SE      INC    RES
0432 20    DD               BRA    SETUP
0434 B6    0467   OTHER     LDA    RES
0437 BB    0466             ADDA   INTER
043A B7    0467             STA    RES
043D 7C    0465             INC    COUNTR
0440 B6    E00A             LDA    SECCNT
0443 B1    0465             CMPA   COUNTR
0446 26    C9               BNE    SETUP
0448 39                     RTS

0449 0A0D         INTMES    FDB    $0A0D
044B 49 4E 50 55            FCC    'INPUT INTERLEAVE FACTOR: ',4
0465              COUNTR    RMB    1
0466              INTER     RMB    1
0467              RES       RMB    1
0468              TRKEND    RMB    2

                  * DRIVE SELECT ROUTINE

046A A6    03     SELECT    LDA    3,X
046C 81    01               CMPA   #1
046E 22    05               BHI    VALDRV
0470 C6    0F               LDB    #$F
0472 1A    01               SEC
0474 39                     RTS
0475 81    03     VALDRV    CMPA   #3
0477 27    09               BEQ    ONE
0479 86    FF               LDA    #$FF
047B B7    E00E             STA    WSDH
047E 5F                     CLRB
047F 1C    FE               CLC
0481 39                     RTS
0482 86    FE     ONE       LDA    #$FE
0484 B7    E00E             STA    WSDH
0487 5F                     CLRB
0488 1C    FE               CLC
048A 39                     RTS

          00E0              SETDP  HBASE/256 THIS IS NEEDED TO SUPPRESS WARNINGS AT ASSEMBLY

                  BLOCK     MACRO  *          THIS IS A FAST WAY TO BLOCK-MOVE DATA FROM THE
                            LDA    <RDATA     WD1000 TO THE FLEX SECTOR BUFFER.  HOWEVER, IT
                            LDB    <RDATA     IS REPEATED 64 TIMES AND REALLY CHEWS UP A LOT
                            STD    X++        OF MEMORY.  SOME MAY WISH TO CODE A LOOP INSTEAD
                            ENDM   *          TO SAVE MEMORY AT THE COST OF SLOWER TRANSFER

                  * READ A SECTOR
                  * A HAS TRACK, B HAS SECTOR, X HAS BUFFER ADDRESS

          000F    READ      EQU    $F
          000F    WRITE     EQU    $F
048B 34    08     READC     PSHS   DP         SAVE DIRECT PAGE
048D 43                     COMA
048E 34    02               PSHS   A          SAVE TRACK TEMPORARILY
0490 5A                     DECB              FIX SECTOR FOR FLEX
0491 53                     COMB
0492 86    E0               LDA    #HBASE/256 GET DIRECT PAGE OF DISK INTERFACE
0494 1F    8B               TFR    A,DP
0496 35    02               PULS   A          RECOVER TRACK
0498 D7    0B               STB    <WSNUMB    GIVE IT THE SECTOR
049A 97    0C               STA    <CYLL      AND THE TRACK
049C 86    0F               LDA    #READ      ISSUE A "READ SECTOR" COMMAND
049E 97    0F               STA    <COMAND
04A0 0D    0F     READ2     TST    <STATUS
04A2 2A    FC               BPL    READ2      BUSY?
04A4 96    0F     READ3     LDA    <STATUS
04A6 85    10               BITA   #$10       SEEK COMPLETE?
04A8 26    FA               BNE    READ3
          04AA    HREAD     EQU    *

                  * THE ASSEMBLY DOESN'T SHOW NEXT LINE WHICH IS "RPT 64" IN SOURCE
04AA                        BLOCK  *          HERE IS WHERE MEMORY IS REALLY USED UP

                  * "RPT 64" AGAIN
052A                        BLOCK

05AA D6    0F               LDB    <STATUS
05AC C5    01               BITB   #1         ERROR?
05AE 27    04               BEQ    READ8
05B0 C6    00               LDB    #0
05B2 35    88               PULS   DP,PC      RETURN WITH NO ERRORS FOUND
05B4 96    09     READ8     LDA    <ERROR     ERROR, CHECK IT
05B6 81    EF               CMPA   #%10111111 DATA CRC ERROR?
05B8 26    04               BNE    RERR6
05BA C6    08               LDB    #8         MUST BE CRC
05BC 35    88               PULS   DP,PC
05BE C6    10     RERR6     LDB    #$10       10 ERROR FOR FLEX
05C0 35    88               PULS   DP,PC

                  WBLOCK    MACRO  SEE        COMMENTS UNDER "BLOCK"
                            LDD    X++
                            STA    <WDATA
                            STB    <WDATA
                            ENDM

                  * ROUTINE TO WRITE OUT A SECTOR TO DISK

05C2 34    08     WRITEC    PSHS   DP         THIS WORKS JUST ABOUT LIKE "READC"
05C4 43                     COMA
05C5 34    02               PSHS   A
05C7 5A                     DECB
05C8 53                     COMB
05C9 86    E0               LDA    #HBASE/256
05CB 1F    8B               TFR    A,DP
05CD 35    02               PULS   A
05CF D7    0B               STB    <WSNUMB
05D1 97    0C               STA    <CYLL
05D3 86    0F               LDA    #WRITE
05D5 97    0F               STA    <COMAND
05D7 0D    0F     WRIT2     TST    <STATUS
05D9 2A    FC               BPL    WRIT2
05DB 96    0F     WRIT3     LDA    <STATUS
05DD 85    10               BITA   #$10
05DF 26    FA               BNE    WRIT3
          05E1    HWRIT     EQU    *

                  * SOURCE HAS "RPT 64" ON NEXT LINE WHICH DOESN'T SHOW AT ASSEMBLY
05E1                        WBLOCK *          AGAIN, WE CHEW UP MEMORY WITH THIS SEQUENCE

                  * ANOTHER "RPT 64" HERE
0661                        WBLOCK

06E1 0D    0F     WRIT4     TST    <STATUS
06E3 2A    FC               BPL    WRIT4
06E5 D6    0F               LDB    <STATUS
06E7 C5    01               BITB   #1         ERROR?
06E9 26    04               BNE    WRIT7
06EB C6    10               LDB    #%00010000 MUST BE DATA FIELD
06ED 20    04               BRA    WRIT8
06EF C6    00     WRIT7     LDB    #0
06F1 1C    FE               CLC
06F3 35    88     WRIT8     PULS   DP,PC
          06F5    SECTAB    EQU    *
                            END    START

0 ERROR(S) DETECTED
```

```
****************************************************************
* OVERLAY FOR FLEX FLOPPY DISK DRIVERS AND HARD DISK DRIVER    *
* ROUTINES (CO-RESIDENT) ORIGINALLY WRITTEN AS STAND-ALONE     *
* WINCHESTER DRIVERS BY R. ZEFF, 5/26/82.  NEW VERSION BY D.J. *
* GRAVES, 7/13/82.  THESE DRIVERS DO NOT CHECK FOR WRITE       *
* ERRORS, WHICH ACCORDING TO ZEFF SLOWS RESPONSE DRAMATICALLY. *
* THE FLOPPIES RETAIN THEIR ORIGINAL NUMBERING OF 0 AND 1 AND  *
* THE WINCHESTER IS CONFIGURED AS TWO DRIVES, 2 AND 3.         *
* DAVID J. GRAVES (215) 898-7951                               *
* 311A TOWNE BLDG./D3                                          *
* UNIV. OF PENNA.                                              *
* PHILA. PA. 19104                                             *
****************************************************************

CD03  WARMS   EQU  $CD03     THESE ARE ALL STANDARD FLEX EQUATES
CD39  OUTDEC  EQU  $CD39
CD15  GETCHR  EQU  $CD15
CD1B  INBUFF  EQU  $CD1B
CD33  SETEXT  EQU  $CD33
D406  FMS     EQU  $D406
CD2D  GETFIL  EQU  $CD2D
CD18  PUTCHR  EQU  $CD18
CD0F  OUTCH   EQU  $CD0F
CD1E  PSTRNG  EQU  $CD1E
CD24  PCRLF   EQU  $CD24
CD48  INDEC   EQU  $CD48
CD42  GETHEX  EQU  $CD42
CD45  OUTADR  EQU  $CD45
CD4E  START   EQU  $CD4E
CD3F  RPTERR  EQU  $CD3F
CD3C  OUTHEX  EQU  $CD3C
CD27  NXTCH   EQU  $CD27
CC0E  DATE    EQU  $CC0E
CD68  WARGO   EQU  $CD68

DE31  FREAD   EQU  $DE31     THESE ARE THE PRESENT ADDRESSES OF FLEX
DEA2  FWRITE  EQU  $DEA2     DISK DRIVERS IN MY SYSTEM AND WILL DEPEND
DED9  FVERIF  EQU  $DED9     ON FLEX VERSION IN USE.
DEF3  FRESTR  EQU  $DEF3
DF19  FDRIVE  EQU  $DF19
DF41  FCHECK  EQU  $DF41
DE26  FINIT   EQU  $DE26

E008  HBASE   EQU  $E008     SYSTEM DEPENDENT: FIRST ADDRESS OF DISK PORT
E008  RDATA   EQU  HBASE     REGISTER 1 FOR READ PURPOSES
E010  WDATA   EQU  HBASE+8   REGISTER 1 FOR WRITE PURPOSES
```

```
E031 ERROR  EQU   HBASE+1  REGISTER 2, ETC. FOLLOW IN ORDER
E039 WPRE   EQU   HBASE+9  SEE WD1000 MANUAL FOR DEFINITION OF REGISTERS
E032 RSCNT  EQU   HBASE+2
E03A WSCNT  EQU   HBASE+10
E033 RSNUMB EQU   HBASE+3
E03B WSNUMB EQU   HBASE+11
E034 RCYLL  EQU   HBASE+4
          E03C WCYLL  EQU   HBASE+12
          E035 RCYLH  EQU   HBASE+5
          E03D WCYLH  EQU   HBASE+13
          E036 RSDH   EQU   HBASE+6
          E03E WSDH   EQU   HBASE+14
          E037 STATUS EQU   HBASE+7
          E03F COMAND EQU   HBASE+15

          0007 BELL   EQU   7          ASCII CONTROL G FOR BELL SIGNAL
          E004 PORT   EQU   $E004      TERMINAL PORT ADDRESS
          0020 SECMAX EQU   32
          00FF TRKMAX EQU   255
          00AF FORMAT EQU   %00101111  NOTE THAT THIS AND ALL OTHER COMMANDS MUST
          FFDF PRECMP EQU   !32        BE GIVEN AS THEIR COMPLEMENT SINCE DATA IS
          00EB RESTSL EQU   $EB        NOT RE-INVERTED BY THE 74LS245 DRIVER
          00EF REST   EQU   $EF        (TRUE FOR REGISTER DATA TOO)

                 * OVERLAY FOR FLEX DISK ENTRY POINTS

CD03                     ORG   $CD03
CD03 7E  E801    SETUP   JMP   HDWARM   MAKE SURE WINCHESTER IS INITIALIZED

DE00                     ORG   $DE00

DE00 7E  E807    READ1   JMP   HREAD    CHECK DATA AT DE00,DE04,DE07....
DE03 7E  E81A    WRITE1  JMP   HWRITE   AND IF NECESSARY CHANGE THE CORRESPONDING
DE06 7E  E836    VERIFY  JMP   HVERIF   EQUATES (I.E. FOR READ1 CHANGE FREAD, ETC.)
DE09 7E  E843    RESTR1  JMP   HRESTR
DE0C 7E  E84A    DRIVE   JMP   HDRIVE
DE0F 7E  E851    CHECK   JMP   HCHECK
DE12 7E  E858    QUICK   JMP   HQUICK
DE15 7E  E85A    INIT    JMP   HINIT

E800                     ORG   $E800    RAM TO STORE THE DRIVERS IS ASSUMED HERE

E800             HRDFLG  RMB   1

                 * INITIALIZE DRIVE AT WARMSTART

E801 17  0073    HDWARM  LBSR  RESTRE
E804 7E  CD4B            JMP   WARMGO   AND RETURN TO FLEX WARMSTART ROUTINE

                 * FIRST WE CHECK FOR HARD OR FLOPPY DISK AND BRANCH APPROPRIATELY

E807 30  88 C0   HREAD   LEAX  -64,X    MOVE BACK TO START OF FCB
E80A 8D  53              BSR   DRVNUM   SUBROUTINE CHECKS TYPE OF DRIVE
E80C 2C  06              BGE   HARD1
E80E 30  88 40           LEAX  64,X     NOW BACK TO SECTOR BUFFER
E811 7E  DE31            JMP   FREAD    JUMP TO OLD FLOPPY ROUTINE
E814 30  88 40   HARD1   LEAX  64,X
E817 16  00BF            LBRA  READC    DO HARD DISK ROUTINE
E81A 30  88 C0   HWRITE  LEAX  -64,X    DO THE SAME FOR ALL OTHER FUNCTIONS
E81D 8D  40              BSR   DRVNUM
E81F 2C  09              BGE   HARD2

E821 30  88 40           LEAX  64,X
E824 7F  E800            CLR   HRDFLG   WHEN CLEAR, FLOPPY WRITE
E827 7E  DE?C            JMP   FWRITE
E82A 30  88 40   HARD2   LEAX  64,X
E82D 7F  E800            CLR   HRDFLG
E830 73  E800            COM   HRDFLG   WHEN SET, HARD DISK WRITE
E833 16  0?C9            LBRA  WRITEC
E836 7D  E800    HVERIF  TST   HRDFLG
E839 26  05              BNE   HDCHK
E83B 7E  DEF9            JMP   FVERIF
E83E C6  00      HDCHK   LDB   #0       DON'T CHECK FOR WRITE ERRORS
E840 1C  FE              CLC
E842 39                  RTS
E843 8D  1A      HRESTR  BSR   DRVNUM
E845 2C  21              BGE   DREST
E847 7E  DEF3            JMP   FRESTR
E84A 8D  13      HDRIVE  BSR   DRVNUM
E84C 2C  4F              BGE   HDRIV
E84E 7E  DF19            JMP   FDRIVE
E851 8D  0C      HCHECK  BSR   DRVNUM
E853 2C  1D              BGE   DUMCHK
E855 7E  DF41            JMP   FCHECK
E858 20  F7      HQUICK  BRA   HCHECK   SAME AS DRIVE CHECK
E85A 8D  1B      HINIT   BSR   RESTRE
E85C 7E  DE26            JMP   FDINIT

E85F 34  02      DRVNUM  PSHS  A        SAVE A
E861 A6  03              LDA   3,X      GET DRIVE # FROM FCB+3
E863 81  02              CMPA  #2       HARD OR FLOPPY DISK?
E865 35  02              PULS  A        RECOVER A
E867 39                  RTS

E868 86  07      DREST   LDA   #BELL    BEEP TERMINAL ON ERROR
E86A BD  CD18            JSR   PUTCHR
E86D C6  00              LDB   #0       DRIVE DOES ITS OWN RECOVERY
E86F 1C  FE              ANDCC #$FE
E871 39                  RTS

E872 C6  00      DUMCHK  LDB   #0       JUST A DUMMY DRIVE CHECK
E874 1C  FE              CLC
E876 39                  RTS

                 * INITIALIZE THE DRIVE

E877 86  FF      RESTRE  LDA   #$FF
E879 B7  E03E            STA   WSDH     CHOOSE DRIVE,SIDE,HEAD AS ZERO
E87C 86  DF              LDA   #PRECMP
E87E B7  E039            STA   WPRE     SET PRECOMPENSATION TRACK
E881 86  FF              LDA   #-1
E883 B7  E03D            STA   WCYLH    CYLINDER HIGH IS ALWAYS ZERO ON 5 MEG DRIVE
E886 86  EB      REST1   LDA   #RESTSL
E888 B7  E03F            STA   COMAND   DO A SLOW RESTORE
E88B 7D  E037    REST3   TST   STATUS
E88E 2A  FB              BPL   REST3    CHECK BUSY FLAG

E890 86  EF              LDA   #REST
E892 B7  E03F            STA   COMAND   NOW DO DUMMY FAST RESTORE TO RESET SEEK SPEED
E895 7D  E037    REST2   TST   STATUS
E898 2A  FB              BPL   REST2    BUSY?
E89A C6  00              LDB   #0
E89C 39                  RTS

                 * DRIVE SELECT ROUTINE

E89D A6  03      HDRIV   LDA   3,X
E89F 81  03              CMPA  #3
E8A1 23  05              BLS   VALDRV
E8A3 C6  0F              LDB   #$F      ERROR RETURN
E8A5 1A  01              SEC
E8A7 39                  RTS
E8A8 27  09      VALDRV  BEQ   ONE
E8AA 86  FF              LDA   #$FF     DRIVE 2 IS TRANSLATED TO HEAD ZERO
E8AC B7  E03E            STA   WSDH
E8AF 5F                  CLRB
E8B0 1C  FE              CLC
E8B2 39                  RTS
E8B3 86  FE      ONE     LDA   #$FE     DRIVE 3 IS TRANSLATED TO HEAD ONE
E8B5 B7  E03E            STA   WSDH
E8B8 5F                  CLRB
E8B9 1C  FE              CLC
E8BB 39                  RTS

          00E0           SETDP HBASE/256 THIS IS NEEDED TO SUPPRESS WARNINGS AT ASSEMBLY

                 BLOCK   MACRO *        THIS IS A FAST WAY TO BLOCK-MOVE DATA FROM THE
                         LDA   <WDATA   WD1000 TO THE FLEX SECTOR BUFFER.  HOWEVER, IT
                         LDB   <WDATA   IS REPEATED 64 TIMES AND REALLY CHEWS UP A LOT
                         STD   X++      OF MEMORY.  SOME MAY WISH TO CODE A LOOP INSTEAD
                         ENDM  *        TO SAVE MEMORY AT THE COST OF SLOWER TRANSFER

                 * READ A SECTOR
                 * A HAS TRACK, B HAS SECTOR, X HAS BUFFER ADDRESS

          00DF   READ    EQU   $DF
          00CF   WRITE   EQU   $CF

                 * FIRST, A COUPLE OF SUBS COMMON TO READC AND WRITEC

E8BC 43          PREDO   COMA
E8BD 34  02              PSHS  A        SAVE TRACK TEMPORARILY
E8BF 5A                  DECB           FIX SECTOR FOR FLEX
E8C0 53                  COMB
E8C1 86  E0              LDA   #HBASE/256 GET DIRECT PAGE OF DISK INTERFACE
E8C3 1F  8B              TFR   A,DP
E8C5 35  02              PULS  A        RECOVER TRACK
E8C7 D7  3B              STB   <WSNUMB  GIVE IT THE SECTOR
E8C9 97  3C              STA   <WCYLL   AND THE TRACK
E8CB 39                  RTS
E8CC 97  3F      POSTDO  STA   <COMAND
E8CE 0D  37      POST2   TST   <STATUS
E8D0 2A  FC              BPL   POST2    BUSY?
E8D2 96  37      POST3   LDA   <STATUS
E8D4 85  10              BITA  #$10     SEEK COMPLETE?
E8D6 26  FA              BNE   POST3
E8D8 39                  RTS

                 * NOW FOR READC PROPER

E8D9 34  08      READC   PSHS  DP       SAVE DIRECT PAGE
E8DB 8D  DF              BSR   PREDO    SET UP FOR A READ COMMAND
E8DD 86  DF              LDA   #READ
E8DF 8D  EB              BSR   POSTDO
          E8E1   HREAD   EQU   *

                 * THE ASSEMBLY DOESN'T SHOW NEXT LINE WHICH IS "RPT 64" IN SOURCE

E8E1                     BLOCK *        HERE IS WHERE MEMORY IS REALLY USED UP

                 * "RPT 64" AGAIN
```

```
E861                    BLOCK
E861 30  89 FF88        LEAX    -256,X      RESET X FOR FLEX
E865 D6  37             LDB     <STATUS
E867 C5  01             BITB    #1          ERROR?
E869 27  06             BEQ     READ8
E86B C6  08             LDB     #8
E86D 1C  FE             CLC
E86F 35  88             PULS    DP,PC       RETURN WITH NO ERRORS FOUND
E871 96  31     READ8   LDA     <ERROR      ERROR, CHECK IT
E873 81  0F             CMPA    #%00011111  DATA CRC ERROR?
E875 26  04             BNE     READ6
E877 C6  08             LDB     #8          MUST BE CRC
E879 35  88             PULS    DP,PC
E87B C6  10     READ6   LDB     #$10        ID ERROR FOR FLEX
E87D 35  88             PULS    DP,PC

                MBLOCK  FDB     *           SEE COMMENTS UNDER "BLOCK"
                        LDD     X++
                        STA     <$DATA
                        STB     <$DATA
                        END

                * ROUTINE TO WRITE OUT A SECTOR TO DISK

E87F 34  88     WRITEC  PSHS    DP          SAVE DIRECT PAGE
E881 17  FDBB           LBSR    FREAD       THIS WORKS JUST ABOUT LIKE "READC"
E884 86  CF             LDA     #WRITE
E886 17  FDC3           LBSR    POSTIO
         E889   MWRIT   EQU     *

                * SOURCE HAS "RPT 64" ON NEXT LINE WHICH DOESN'T SHOW UP HERE

                * ANOTHER "RPT 64" HERE

E889                    MBLOCK
E889 30  89 FF88        LEAX    -256,X
E88D 6D  37     WRIT4   TST     <STATUS
E88F 2A  FC             BPL     WRIT4
E891 D6  37             LDB     <STATUS
E893 C5  01             BITB    #1          ERROR?
E895 26  04             BNE     WRIT7
E897 C6  10             LDB     #$0810      MUST BE DATA FIELD
E899 20  04             BRA     WRIT8
E89B C6  08     WRIT7   LDB     #8
E89D 1C  FE             CLC
E89F 35  88     WRIT8   PULS    DP,PC
                        END     WRMS

0 ERROR(S) DETECTED
```

NOTE: Since the question of 'flakey' 6809 CPU devices was brought up a few months past, in 68 Micro Journal, some additional information has been furnished by Motorola engineers.

In addition to the mask set numbers indicated in the above article, another mask set number "W8L" also has some 'slight' problems with the MRDY function.

The current (for about the past year) mask set number "CW3" has no reported problems with the MRDY function, or any other function, according to Motorola. Guess they finally figured it out.

It is strongly suggested that if you are having **ANY** problem, with your 6809 computer, that is difficult to locate, then I would suggest that you suspect the 6809. Especially if it is not a "CW3" mask set number.

Since the original article appeared I have received letters or telephone calls from over 20 readers stating that having read the article, changed to a new 6809, they have eliminated long time problems. Seems that there is a large number of users of the 6809 that just did not know that Motorola put out a notice in their "Update" publication, concerning these flakey 6809s. We didn't either, or I would have told you sooner!

Come on Motorola, you know that not every user of the 6809 gets your publications. You also know we exist and try to keep our readers aware of such faults. Let me know and I will let thousands of your users know. **Fair enough?**

Actually not every 6809 computer uses the MRDY function, but for those that do, this information should have had more exposure. I wonder how many thousands of wasted hours were spent by 6809 users trying to find the problem. The sad part is that not one manufacturer, that I am aware of notified their dealers or users of this problem, maybe they didn't get the information either. If any of you ever run across something like this again, **PLEASE** let me know, it could save others hours or even months of needless hassle. In this case I am certain it could, according to what I have been told so far.

The Motorola fix includes using the old, and flakey 6809, but adding additional components to the CPU circuit. This of course is a **BAD** approach. First, most current users are not technical types. They are **USERS not FIXERS.** Second, it seems replacement 6809s should have been offered, not instructions for modifying a CPU board. The offer of a replacement 6809 would have been much better (public relations wise) than all the headaches caused by this known "bug" or the "official" fix. Wonder what Intel or the other CPU manufacturers would have done?

The best thing to do, if you suspect your 6809 is to purchase a new one (CW3), plug it in and hope your problem goes away. For some it has. And at current prices for a 6809 (about $15.00) messing around with a CPU board is for the birds.

Sorry I did not get this information sooner but this is being typed within hours of actually getting the "official" poop. Let me know if it helped.

A RUMOR: Look for Motorola to announce a couple of new generation 6809 devices. One should be 3 mhz CMOS 6809 and the other a CMOS 6809 with an expanded instruction set. Should be interesting!

DMW - - -

FLEX LINKED LIST

Have you ever crashed a disk?! I'm talking about linking several files together. You start listing one file out, and suddenly another file appears on the CRT. Horror of horrors - no backup. I always think that next time I will make a backup!!!!! But alas the next time always comes still no backup, usually due to a lack of disks.

This problem happens when a disk file isn't closed properly. FLEX leaves the sector links hanging in limbo until the file is closed, so when you take out the disk while the file is open who knows how the links are setup. This only happens when you're updating or writing to the file (changing it).

Under FLEX the data sectors on the disk are setup in a linked list. That is each sector points to the next sector in the chain.

When the disk is first initialized there is only one file on the disk. This file is the free chain or a linked list of all the available sectors. As each of the available sectors is used, it is linked into the separate file chain, out of the free chain. Now you can see the problem of the file being half completed or not closed. The last sector written still has the old sector link in it which usually points to the free chain. Thus the file is linked into the free chain with all its garbage.

Now you may be wondering what I mean by the terms sector chain and sector links. The disk is formatted into what we call sectors. These sectors are marked by magnetic impulses on the disk, which a special chip can recognize. Under FLEX these sectors are 256 bytes long. This is the only portion of the disk we can use for data. All the other parts of the disk are used

FLEX LINKED LIST

by a special chip for other purposes. FLEX divides up each sector into three different parts. The first two bytes of each sector are used to point to the following sector (this is the sector link information). The next two bytes are used for a logical file sector number. So you can see that FLEX only allows us 252 bytes of data for each sector. The next time you read about XBASIC and it's file structure, notice that its random buffer size is 252 bytes.

As I mentioned before you can see the problems with getting these links messed up. You can have two files mixed together or you can have your file mixed in with the free chain. The way FLEX recognizes the end of a file is by the logical sector number. It equals zero.

The utility at the end of this article is used to print out the current track and sector, the sector link, and the logical file sector number. I've used this utility to recover disks that had irreplacable data on them. It works very well.

The utility is brought up by simply typing its name, followed by a drive number. If no drive number is specified the work drive is used. The utility then asks for the number of tracks on the drive and the number of sectors per track. The default values shown with the prompts are the defaults for a 5 1/4" 35 track drive which I use.

The trick to making the utility work is to follow the sector links and the logical file numbers until the logical file numbers go out of sequence. Where the file has been crashed, simply restore the proper links and your home free. Another good utility to use in conjunction with this is TSC Examine utility.

REFERENCES

FLEX User's Manual

FLEX Advanced Programmer's Guide


```
                                     *********************************
                                     *
                                     * PROGRAM          PRINT SECTOR INFO
                                     *
                                     *                  VER 1.1
                                     *
                                     * WRITTEN BY       KIM MANSFIELD
                                     * FOR              [illegible] COMPUTERS
                                     * DATE             5/24/82
                                     *
                                     * SYNOPSIS         THIS ROUTINE IS THE
                                     *                  EXECUTIVE ROUTINE THAT
                                     *                  PRINTS SECTOR LINK
                                     *                  INFO EITHER ON THE CRT
                                     *                  OR A PRINTER
                                     *
                                     * CALLING PARAMS   NONE
                                     *
                                     * CALLING SPECS    NONE
                                     *
                                     * REGS CHANGED     [illegible] CC
                                     *
                                     * FILE NAME        PNTSCT.TXT
                                     *
                                     **************** CONSTANTS ****************
                                     *
            CC0C                     WORKDR   EQU   X'CC0C'
            CD03                     FLEX     EQU   X'CD03'
            DE0C                     DRIVE    EQU   X'DE0C'
            DE0F                     CHKRDY   EQU   X'DE0F'
            DE09                     RESTOR   EQU   X'DE09'
            DE00                     READ     EQU   X'DE00'
            CD24                     PCRLF    EQU   X'CD24'
            CD18                     PUTCHR   EQU   X'CD18'
            CD27                     NXTCH    EQU   X'CD27'
            CD1E                     PSTRNG   EQU   X'CD1E'
            CD1B                     INBUFF   EQU   X'CD1B'
            CD42                     GETHEX   EQU   X'CD42'
            CD3F                     RPTERR   EQU   X'CD3F'
                                     *
                                     *************** EXTERNAL ROUTINES *******
                                     *
            0000                     PRINTIT  EXT
                                     *
                                     *****************************************
                                     *
            0000                     PNTSCT   ENT
     3      0000   20 05    (0007)            BRA   START
            0002 01280 1                      CON   X'01',X'28',X'01' VERSION 0
     6      0005   16 0100  (0108)   START    LBRA  ENDTABLE
            0008 5345435420 4C       MESS1    CON   C'SECT LINK FILE        ',0
            001C 3030303020 3D       MESS2    CON   C'---- ---- ----        ',0
            0030 454E544552 20       MESS3    CON   C'ENTER HEXADECIMAL NUMBER '
```

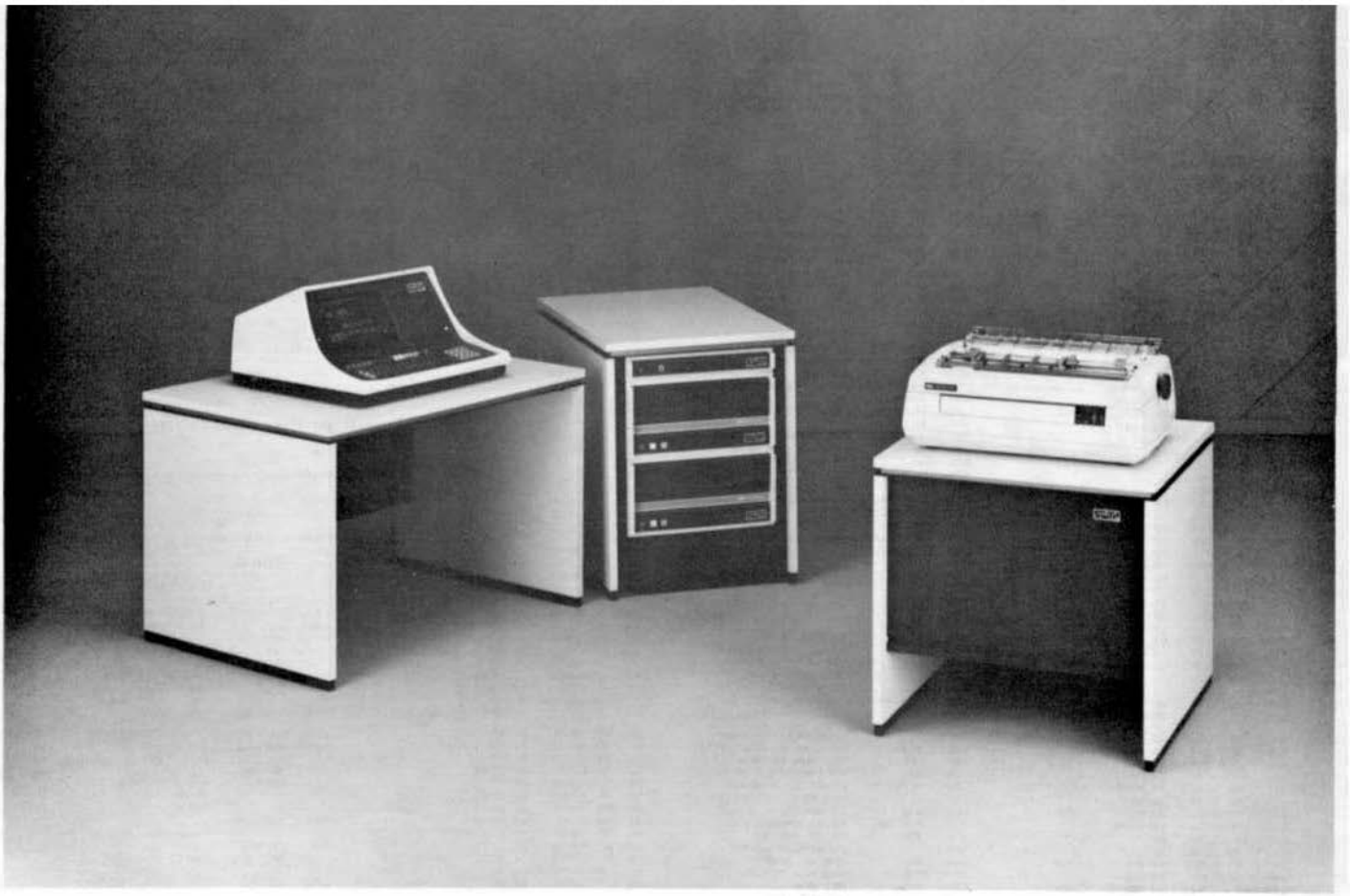
```
      0049 464F52205448          FCC    C'FOR THE MAXIMUM NUMBER OF '
      0063 545241434B53          FCC    C'TRACKS = 1',X'04'
      006E 284445464155  MESS4   FCC    C'(DEFAULT IS 0221? ',X'04'
      0081 454E54455220  MESS5   FCC    C'ENTER HEXADECIMAL NUMBER '
      009A 464F52205448          FCC    C'FOR THE MAXIMUM NUMBER OF '
      00B4 534543544F52          FCC    C'SECTORS ',X'04'
      00BD 284445464155  MESS6   FCC    C'(DEFAULT IS 00A)? ',X'04'
      00D0 22            MAXTRK  FCC    X'22'
      00D1 0A            MAXSCT  FCC    X'0A'
      00D2 0000          TRSCTR  FCC    DX'0000'
      00D4 0000          FCB     FCC    DX'0000'
      00D6 0000                  FCC    DX'0000'
      00D8 [illegible]   BUFFER  FCC    256X'00'
11R   01D8  34 7E        ENDTABLE PSHS  A,B,DP,X,Y,U
19   001DA  BD CD27              JSR    NXTCH        GET AFTER COMMAND
21    01DD  81 30                CMPA   #X'30'       DRIVE NUMBER IS IT OK
24    01DF  2D 0E    (01EF)      BLT    @2           GOTO @2 IF NOT
2R    01E1  81 33                CMPA   #X'33'       IS IT OK
5     01E3  2E 0A    (01EF)      BGT    @2           GOTO @2 IF NOT
3R    01E5  80 30                SUBA   #X'30'       MAKE IT BINARY
7     01E7  30 8DFEE4 (0004)     LEAX   FCB,PC       PUT IT IN FCB
12    01EB  A7 03                STA    3,X
15    01ED  20 09    (01F8)      BRA    @3           SKIP DEFAULT
5R    01EF  B6 CC0C      @2      LDA    WORKDR       GET WORK DRIVE FOR
10    01F2  30 8DFEDE (0004)     LEAX   FCB,PC       DEFAULT PUT IT
15    01F6  A7 03                STA    3,X          IN FCB
5R    01F8  30 8DFE34 (0030) @3  LEAX   MESS3,PC     POINT TO MESSAGE
13   001FC  BD CD1E              JSR    PSTRNG
18    01FF  30 8DFE6B (006E)     LEAX   MESS4,PC     POINT TO MESSAGE
26   00203  BD CD1E              JSR    PSTRNG
34   00206  BD CD1B              JSR    INBUFF       GET NUMBER IN BUFF
42   00209  BD CD42              JSR    GETHEX       CONVERT IT TO HEX
45    020C  25 EA    (01F8)      BCS    @3           CS BAD NUMBER
6R    020E  1F 10                TFR    X,D          PUT IT IN D
9     0210  27 04    (0216)      BEQ    @35          IF NOTHING DEFAULT
5R    0212  E7 8DFEBA (00D0)     STB    MAXTRK,PC    MAKE IT MAXTRK
5R    0216  30 8DFE67 (0081) @35 LEAX   MESS5,PC     POINT TO MESSAGE
13   0021A  BD CD1E              JSR    PSTRNG
18    021D  30 8DFE9C (00BD)     LEAX   MESS6,PC     POINT TO MESSAGE
26   00221  BD CD1E              JSR    PSTRNG
34   00224  BD CD1B              JSR    INBUFF
42   00227  BD CD42              JSR    GETHEX
45    022A  25 EA    (0216)      BCS    @35
6R    022C  1F 10                TFR    X,D
9     022E  27 04    (0234)      BEQ    @15
5R    0230  E7 8DFE9D (00D1)     STB    MAXSCT,PC
8R   00234  BD CD24      @15     JSR    PCRLF        OUTPUT RETURN
10    0237  C6 04                LDB    #4           PRINT FOUR MESSAGES
3R    0239  30 8DFDCD (0008) @10 LEAX   MESS1,PC     POINT TO MESSAGE
6R    023D  A6 80        @20     LDA    0,X+         GET NEXT CHAR OF MESS
9     023F  27 05    (0246)      BEQ    @30          IF MSG DONE
8R   00241  BD CD18              JSR    PUTCHR       PRINT CHARACTER
11    0244  20 F7    (023D)      BRA    @20          DO IT AGAIN

2R    0246  5A           @30     DECB                DECREMENT COUNTER
5     0247  26 F0    (0239)      BNE    @10          HAVE WE PRINTED FOUR
8R   00249  BD CD24              JSR    PCRLF        YES NOW RETURN
10    024C  C6 04                LDB    #4           ANOTHER FOUR MESS
3R    024E  30 8DFDCA (001C) @40 LEAX   MESS2,PC     POINT TO MESSAGE
6R    0252  A6 80        @50     LDA    0,X+         GET NEXT CHARACTER
9     0254  27 05    (025B)      BEQ    @60          IF MSG DONE
8R   00256  BD CD18              JSR    PUTCHR       PRINT CHARACTER
11    0259  20 F7    (0252)      BRA    @50          DO IT AGAIN
2R    025B  5A           @60     DECB                DECREMENT COUNTER
5     025C  26 F0    (024E)      BNE    @40          ARE WE DONE?
8R   0025E  BD CD24              JSR    PCRLF        YES NOW RETURN
13    0261  30 8DFE9F (0004)     LEAX   FCB,PC       GET FCB
21   00265  BD DE0C              JSR    DRIVE        SELECT DRIVE
5R    0268  30 8DFE98 (0004) @1  LEAX   FCB,PC       GET FCB
13   0026C  BD DE0F              JSR    CHKRDY       DRIVE READY
16    026F  26 F7    (0268)      BNE    @1           NO
5R    0271  30 8DFE8F (0004)     LEAX   FCB,PC       YES GET FCB
13   00273  BD DE09              JSR    RESTOR       HOME HEAD
16    0278  8E 0001              LDX    #X'0001'     START TRK 0 SECT 1
22    027B  AF 8DFE53 (00D2)     STX    TRSCTR,PC    SAVE IT
6R    027F  EC 8DFE4F (00D2) @5  LDD    TRSCTR,PC    GET NEXT TRK SCT
11    0283  30 8DFE51 (00D8)     LEAX   BUFFER,PC    GET BUFFER ADDRESS
19   00287  BD DE00              JSR    READ         READ SECTOR
22    028A  26 2A    (02B6)      BNE    ERROR        GO IF ERROR
6R    028C  AE 8DFE42 (00D2)     LDX    TRSCTR,PC    GET TRK SCT
11    0290  31 8DFE44 (00D8)     LEAY   BUFFER,PC    POINT TO BUFFER
18    0294  34 30                PSHS   X,Y          PARAMETERS
25   00296  17 FD67  (0000)      LBSR   PRINTIT      PRINT INFO
32    0299  35 30                PULS   X,Y          CLEAR STACK
38    029B  1F 10                TFR    X,D
43    029D  E1 8DFE30 (00D1)     CMPB   MAXSCT,PC    MAX SECTOR
46    02A1  26 09    (02AC)      BNE    @4           NO
2R    02A3  4C                   INCA                YES INC TRK
7     02A4  A1 8DFE28 (00D0)     CMPA   MAXTRK,PC    MAX TRACK
10    02A8  27 09    (02B3)      BEQ    DONE2        YES DONE
2R    02AA  C6 00                LDB    #X'00'       NO SET SECT 1
2R    02AC  5C           @4      INCB
8     02AD  ED 8DFE21 (00D2)     STD    TRSCTR,PC    SAVE TRK SCTR
11    02B1  20 CC    (027F)      BRA    @5           DO IT AGAIN
4R    02B3  7E CD03      DONE2   JMP    FLEX         FINISHED
2R    02B6  86 09        ERROR   LDA    #9           DISK ERROR #9
7     02B8  30 8DFE18 (0004)     LEAX   FCB,PC       PUT IN FCB
12    02BC  A7 01                STA    1,X
20   002BE  BD CD3F              JSR    RPTERR       REPORT ERROR
24    02C1  7E CD03              JMP    FLEX         GOTO FLEX
      02C4                       END    PNTSCT

0008 BUFFER
DE0F CHKRDY
02B3 DONE2
DE0C DRIVE
01D8 ENDTABLE
02B6 ERROR
0004 FCB
CD03 FLEX
CD42 GETHEX
CD1B INBUFF
00D1 MAXSCT
00D0 MAXTRK
0008 MESS1
001C MESS2
0030 MESS3
006E MESS4
0081 MESS5
00BD MESS6
CD27 NXTCH
CD24 PCRLF
0000 PNTSCT
0000 PRINTIT
CD1E PSTRNG
CD18 PUTCHR
DE00 READ
DE09 RESTOR
CD3F RPTERR
0003 START
0002 TRSCTR
CC0C WORKDR
```

```
                                 ************************************************
                                 *
                                 * PROGRAM        PRINT SECTOR INFORMATION
                                 *
                                 * WRITTEN BY     KIM MANSFIELD
                                 * FOR            CORVIN COMPUTERS
                                 * DATE           5.24.82
                                 *
                                 * SYNOPSIS       THIS ROUTINE PRINTS THE INFO
                                 *                GATHERED BY OTHER ROUTINES
                                 *                WHICH IS THE SECTOR LINKS
                                 *
                                 * CALL PARMS     FIRST PUSH BUFFER ADDRESS
                                 *                NEXT PUSH TRACK AND SECTOR
                                 *                INFO TRACK = MSB SECTOR = LSB
                                 *
                                 * CALL SPECS     TRCKSCTR (0,S)
                                 *                BUFFER   (2,S)
                                 *
                                 * REGS CHANGED   CHANGES CC
                                 *
                                 * FILE NAME      PRINTIT.TXT
                                 *
                                 ***************** CONSTANTS *****************
                                 *
      CD18                       PUTCH    EQU  X'CD18'
      CD24                       PCRLF    EQU  X'CD24'
      CD45                       OUTADR   EQU  X'CD45'
                                 *
                                 ***************** VARIABLES *****************
                                 *
      0000                       STACKSIZ EQU  0
                                 *
                                 ***************** PARAMETERS ****************
                                 *
      0000                       TRCKSCTR EQU  STACKSIZ+1)
      0000                       BUFFER   EQU  STACKSIZ+1)
                                 *
                                 *********************************************
      0000                       PRINTIT  ENT
3     0000  20 07    (0009)               BRA  ENDTABLE
      0002 00                    EOL      FCC  X'00'        ITEM COUNT
      0003 2020202020000         SPACE    FCC  C'     ',X'00' SPACE CONSTANT
11R   0009  34 7E                ENDTABLE PSHS A,B,DP,X,Y,U
15    000B  32 E4                         LEAS -STACKSIZ,S ROOM FOR VARS
19    000D  33 E4                         LEAU 0,S          POINTER IN U
24    000F  A6 8CF0  (0002)               LDA  EOL,PC       GET ITEM COUNT
26    0012  81 04                         CMPA #4           FOUR ITEMS YET?
29    0014  26 06    (001C)               BNE  @1           NO
7R    0016  6F 8CE9  (0002)               CLR  EOL,PC       YES RESET COUNT
15   00019  BD CD24                       JSR  PCRLF        CARRIAGE RETURN
2R    001C  4C                   @1       INCA              INC ITEM COUNT
7     001D  A7 8CE2  (0002)               STA  EOL,PC       SAVE IT

12    0020  30 4D                         LEAX TRCKSCTR,U POINTER TO TRACK SEC
20   00022  BD CD45                       JSR  OUTADR       PRINT IT
22    0025  86 20                         LDA  #X'20'       PRINT SPACE
30   00027  BD CD18                       JSR  PUTCH
36    002A  AE 4D                         LDX  BUFFER,U     POINTER TO BUFFER
44   0002C  BD CD45                       JSR  OUTADR       PRINT SECTOR LINK
46    002F  86 20                         LDA  #X'20'       PRINT SPACE
54   00031  BD CD18                       JSR  PUTCH
59    0034  30 01                         LEAX 1,X          POINT TO LOGICAL NUMB
67   00036  BD CD45                       JSR  OUTADR       PRINT IT
72    0039  31 8CC7  (0003)               LEAY SPACE,PC     PRINT SPACES
6R    003C  A6 A0                @2       LDA  0,Y+
9     003E  27 05    (0045)               BEQ  @3
8R   00040  BD CD18                       JSR  PUTCH
11    0043  20 F7    (003C)               BRA  @2
6R    0045  32 E4                @3       LEAS STACKSIZ,S CLEAR STACK
15    0047  35 7E                         PULS A,B,DP,X,Y,U
20    0049  39                            RTS               RETURN
      004A                                END

0000 BUFFER
0009 ENDTABLE
0002 EOL
CD45 OUTADR
CD24 PCRLF
0000 PRINTIT
CD18 PUTCH
0003 SPACE
0000 STACKSIZ
0000 TRCKSCTR
```

FD88 DEV. SYS.

FD88 Firmware Development System

A few months ago the 68 Micro Journal Labs received a new kind of Standard S50 Bus circuit board. Something that a lot of you can use, especially since we have progressed(?) to the point that two monitors are required to run all our software, ie: FLEX™, OS9™, UniFLEX™ or SDOS™, etc. With the Acorn Computer System FD88 development board any two systems can co-reside on a single 6809 computer at one time. To switch from one to the other a switch located on top of the board switches from one monitor to the other. Or the switch can be mounted outboard and the cover of the computer not messed with, or you can switch by *PIA control. Not only can that be accomplished but provision are made to have both ROM, EPROM and RAM located at the same addresses and then switch by switch or *PIA control copy EPROM or ROM into RAM and then modify a monitor (or any other ROM type program) until all the bugs are worked out. Should worse come to worse (don't it always?) you can switch back to ROM and start all over, nothing lost. Neat, huh?



THE COMPLETE BUSINESS SYSTEM

+Multiuser+Highly Expandable+Cost Effective

S+ THE CONCEPT

The S+ system is a modular computer system in which all portions of the hardware and software are designed to work together in the most efficient way possible. An S+ single user system with floppy disk storage is a competitive and cost effective entry level system. Unlike most other small computers being sold as "personal", or "small business" machines, the S+ system may be expanded to maximum capabilities using this same hardware and software. You cannot end up with a DEAD END system that cannot be expanded and whose software is not compatible with larger machines. A basic S+ system may be expanded to thirty-two users, a megabyte of main memory and hundreds of megabytes of hard disk storage by simply plugging in, or connecting the desired upgrade equipment.

TOTAL DESIGN–Hardware and Software

The S+ system is an integrated hardware and software design. The two complement and enhance each other in this system. The UniFLEX® operating system used in the S+ systems is patterned after the Bell Laboratories UNIX® operating system, one of the most admired and widely used operating systems in the world. Instead of being an afterthought, the software is part of the design of the S+ system. You can be sure that with this approach that all parts of the computer operate with maximum efficiency and cost effectiveness.

THE CENTRAL PROCESSOR

The basic S+ system is configured with 256K bytes of memory and can be expanded to more than 1 million bytes. An efficient and fast hardware memory management system is used to allocate the available memory among the users on a dynamic basis. As little as 8K bytes, or the entire memory–if needed–can be used by any individual user. This makes it possible to run very large programs on the system, but it also uses no more memory than necessary for a particular job. The increase in cost effectiveness of this system over crude and outdated bank switching arrangements is dramatic.

The central processor runs in both user and supervisor states. It can detect and reject a defective user program. It is impossible for a user program to go bad and stop the entire system, as can happen quite easily in less sophisticated systems.

Task switching is accomplished by use of a multiple map RAM memory, with sixty-four individual task maps. Each task can access from 4 to 64 K-bytes of memory. Multiple tasks may be used in programs that require more than 64K bytes of memory for execution. When a task is completed the memory is automatically released for other use.

SOFTWARE

The S+ operating system, UniFLEX® is a multiuser, multitasking operating system based on the UNIX® operating system that has been used for many years on Digital Equipment Corp. PDP-11 series minicomputers. It is considered one of the most sophisticated and "user friendly" operating systems available. Variations of UNIX® are rapidly becoming standard on mini and larger microcomputers.

A large variety of languages are available for use with the system. These include FORTRAN, COBOL, BASIC, and Pascal. Word processing packages are also available to give you full text processing capability on the system.

Applications programs are available in large quantities in many fields. This includes general business, medical, dental, veterinary, library and real estate management; plus others. Since the system is multiuser it can also be connected to cash registers to produce a point-ofsale terminal system combined with the computer. The possibilities for application of this system are endless.

THE I/O SYSTEM

The S+ system is totally interrupt driven. All terminal and printer I/O devices connect to an I/O bus separate from the main bus. Up to thirty-two separate devices may be connected to the I/O bus at any one time. If I/O activity is great enough to cause an unacceptable slowdown in system operation, a separate I/O processor can be installed in the system. This plug-in option removes all I/O handling overhead from the main processor and allows operation of up to thirty-two external devices at 9,600 baud. Without an integrated total design, as in the S+ system, it would become impractical to use a UNIX®type operating system in a situation with heavy terminal I/O activity.

DISK STORAGE

A wide range of disk storage capacity is available for the S+ system, from 2.5 M-byte floppy disks to an 80 M-byte Winchester and many sizes between. All disk controllers use direct memory access (DMA) type operations to maximize data transfer and to minimize overhead on the main processor. The Winchester disks also use intelligent controllers along with DMA transfers to preserve the performance that these type devices are capable of giving. Without this distributed intelligence the system performance would be greatly degraded. The UniFLEX® operating system is designed to work at maximum efficiency with this type disk system. The data transfer rates achieved by this combination rival those of large minicomputers.

COMMUNICATIONS

A high speed local network communications system is available to interconnect S+ systems. The VIABUS® network will allow communication between systems at data rates of over 400K baud. Such a system makes it possible to share data between local systems in an efficient and low-cost manner.

AVAILABLE SOON

Tape backup--20M-Byte in less than 15 minutes on a standard ¼ inch cartridge.

Mini-Wini—5 and 10 M-Byte Winchesters—5¼ inch package. Winchester performance, for smaller systems in a small package. UniFLEX® compatible design.

Large Capacity—190 and 340 M-Byte Winchesters, plus SMD cartridge drives.

UniFLEX is a registered trademark of Technical Systems Consultants, Inc.

UNIX is a registered trademark of Bell Labs.

VIABUS is a registered trademark of Southwest Technical Products Corporation.

SOUTHWEST TECHNICAL PRODUCTS CORPORATION
219 W. RHAPSODY
SAN ANTONIO, TEXAS 78216 **(512) 344-0241**

SOME BACKGROUND

Acorn Computer Systems or ACS started out back in '73. At that time they were a camera service. As most newer cameras have digital circuits they soon found that they were actually in the digital repair business. And like so many of us they soon purchased a SWTPC 6800 computer, for their business. Principles involved are Merle Giesfeldt and Marvin Mess. Both having had many years in the electronic industry. Because they needed special features not found in the standard system monitor they began to design a board that would allow them to design, test and operate any new monitors or other programs they needed. So came the FD88 dual address ROM, EPROM, RAM development board. The first of many new and novel (we are promised) boards for the Standard S50 Bus.

GENERAL

The FD88 was designed to be versatile. Each 2K socket can be jumper-programmed for 2716 (5v) EPROM or 2016-6116 RAM. Each 2K portion may be turned off by switch and removed from effective memory. Each 8K block may be set up on 8K boundries by dip switch, to appear anywhere in the 64K address area. Actually the board is a 16K board and each 8K section can be programmed to the same address space. One block being PROM and the other RAM. Like I said, neat.

By the simple addition of a 74LS138 and 8 pole dip switch the board can be used in extended addressing applications.

By the moving of say FLEX™ into the 8K EPROM portion of the board you will then have all of the FLEX™ subroutines as a portion of your monitor. This is called DOS PRAM-TRANS™ by ACS and gives the user a powerful tool for program development.

OPERATION

The manual is complete and includes information for both the 6800 and 6809 user (something neglected by many). PRAM-TRANS™ is a function (code included) that allows the transfer of data from EPROM to RAM. In fact we are including it in our **new monitor** we are developing for one of our systems. Instructions and hints are include to show how different monitors (GMXBUG™, SWTBUG™, SBUG™, etc.) can be modified. Even hints are given on how to "burn" FLEX™ into EPROM and handle the self-modifying functions of FLEX™ on startup.

Pads are provided on the board for several outboard control switches. By the addition of the furnished short program "DOS PRAM TRANS™" FLEX™, OS9™ or any other program can come up running on your computer with switch on.

The only restrictions to extended address applications, according to the manual, is that if the board is being used for program development, the extended address parts should not be installed. However, if they are installed, and you later decide to use the board for development, a jumper is provided for a switch so that the extended addressing portion can be dissambled.

A 16K RAM - EPROM BOARD

When the FD88 is used as two 8K memory blocks, the switch or (*PIA) is set to address each block separately in the address range. Also each socket may be jumper programmed to be either RAM or EPROM. This allows the board to run in the system as either a 16K RAM or 16K EPROM board.

The FD88 board takes advantage of a "quirk" in the 6116-2016 type RAM chips. If the CS line is low (chip select enabled) and OE is high (output not enabled) you cannot read the chip, but you can write to it.

CONCLUSION

The FD88 development board is a new and welcome addition to the Standard S50 Bus community. It fills a need for a board that will allow users to run two different systems, on the same computer, without messing around and pulling and pushing boards in and out of the computer. In fact, many failures of Standard S50 Bus computers, that I have seen, have been caused by the wearing away of the plating from the connector pins on the 30 or 50 pin bus. Actually, the plating can be removed and the pins damaged to the extend that intermittent problems start to show up after only 15 to 20 removals from the computer, by a particular board. This applies to the tin plated type pins. Those with gold plated (and a lot more expensive) pins can handle up to about 100 or so removals. If you ever look at the circular area of the pin, and the contact surface of the mating socket, it becomes apparent that not much surface area makes contact. Many otherwise good motherboards have been ruined by users who switch between two CPU cards to run different disk systems. The FD88 can eliminate this problem.

The manual is very complete, modeled on the order of Heath™ type manuals it covers everything from soldering tips to parts overlays, board layout, materials list and electronic and logic diagrams. Along with a "how it works" section.

The FD88 is constructed of top grade glass-epoxy, 2 ounce copper and double sided with plated through holes and solder masked.

The price of the bare board is $39.95
kit with parts $84.95
Assembled & tested $114.95

Additional information may be had from:

ASSOCIATED CAMERA SERVICES
11931 W Bluemound Road
Wauwatosa, WISCONSIN 53226
(414) 257-0300

* External

A 68 Micro Journal Lab review - - -

TYPOS

```
/************************* T Y P O S ************************/
/*  Written by: Jodie A. Zoeller, SWTPc, August 1982          */
/*                                                            */
/*  TYPOS is a program designed to help catch spelling errors.*/
/*  It keeps a frequency count of all alphabetic words in the */
/*  specified file. If no tolerance is supplied by the user   */
/*  TYPOS uses 3. This means that all words occurring 3 times */
/*  or less will be displayed after the program is finished   */
/*  scanning the specified file.                              */
/*                                                            */
/*        Syntax:   typos  <file name> [tolerance]            */
/************************************************************/
#include <stdio.h>        /* library of standard routines  */
#include <ctype.h>        /* library with isalpha prog.    */
#define  MAX   100        /* maximum length of buffer      */
#define  CR    0x00       /* hex value for carriage return */
#define  ERR   -1         /* error return code             */
#define  EOF   0          /* end of file return code       */
#define  HYPHEN  '-'      /* hyphen character              */
#define  QUOTE 0x27       /* ' character, single quote     */

typedef struct symbol { struct symbol *left;
                        struct symbol *right;
                        char          *text;
                        long          count;
                      } SYMBOL;
char  word[MAX];          /* buffer to collect word in */
int   infd;               /* file descriptor for input */
SYMBOL  *treeroot = NULL;
```

```
/************************ M A I N ******************************/
main(argc,argv)  int argc; char *argv[];
{ extern SYMBOL *treeroot;
  long t;  /* tolerance flag */
  long getdcml();

if (argc<2) { printf("Syntax: TYPOS  <file name> [tolerance)\n");
              exit(99); }

if ((infd = open(argv[1],0)) == ERR)
   { printf("Unable to open file %s\n",argv[1]);
     exit(99); }

pfilnit();
if (argc>2) { t = getdcml(argv[2]); }
else  t = 3;      /* default value */

if (t==0) exit(0);
scanfile();
close(infd);
/* output section */
printf("\n  T Y P O S  Word Frequency Counter \n");
for (;t>1;t--)
  { printf("\nWords occuring %ld times:\n",t);
    traverse(treeroot,t);  }
printf("\nWords occuring 1 time:\n",t);
traverse(treeroot,t);
} /* end MAIN   */

/*********************** S C A N F I L E ************************/
scanfile()
{ extern  int   infd;
  extern  char  word[MAX];
  char    *origin;            /* pointer of word[0] */
  int     cc;                 /* character count    */

cc = 0;   origin = word;

while ((read(infd,(origin + cc),1)) != 0)
   {  if (isalpha(word[cc])) cc++;
      else if ((word[cc] == QUOTE) && (cc > 0)) cc++;
      else if ((word[cc] == HYPHEN) && (cc > 0)) cc++;
      else if (cc > 0)
         { word[cc] = '\0';
           treeword();
           cc = 0; }
      if (cc == MAX) {
         printf("A word greater then %d characters encountered.\n",cc);
         printf("TYPOS program cannot continue.\n");
         printf("See your utilities manual for TYPOS criteria.\n\n");
         return(1);   }
   }

  return(1);
  }

  /************************ T R E E W O R D ************************/
  treeword()
  { extern SYMBOL *treeroot;
    extern char   word[MAX];
    SYMBOL  *node, *find();

  node = find(word,&treeroot);
  node->count = node->count + 1;
  return(1);
  }

  /*************************** F I N D ****************************/
  SYMBOL *find(token,parent)
    char *token; SYMBOL **parent;
  { int bc;
    SYMBOL  *current, *node();
  while ((current = *parent) != NULL) {
    if ((bc=strcmp(token,current->text))==0)
       return(current);
    else if (bc<0) parent = &(current->left);
    else           parent = &(current->right);
    }
  return (*parent = node(token));
  }

  /*************************** N O D E ****************************/
  SYMBOL *node(key)   char *key;
  { char *cptr; SYMBOL *loc;

  if ((loc = (SYMBOL *) sbrk(sizeof(SYMBOL))) == NULL)
     giveup("Symbol Bucket");
  loc->left =
  loc->right =  NULL;
  loc->count = 0;
  if ((cptr=sbrk(strlen(key)+1))==NULL)
     giveup("Symbol Definition");
  loc->text = cptr;
  strcpy(cptr,key);
  return(loc);
  }

/************************ T R A V E R S E ************************/
traverse(tree,t)  SYMBOL  *tree; long t;
{  SYMBOL *subnode;
   long  temp;
   if (tree == NULL) return;
   if ((subnode = tree->left) != NULL)
      traverse(subnode,t);
   if (tree->count == t) { printf("  %s\n",tree->text); }
   if ((subnode = tree->right) != NULL)
      traverse(subnode,t);
   return;
}

/************************ G E T D C M L ************************/
long getdcml(ptr)  char *ptr;
{ long num;
for (num=0; *ptr>='0' && *ptr<='9'; ptr++)
    num = 10 * num + (*ptr - '0');
return(num);
}

/************************ G I V E U P ************************/
giveup(what)    char *what;
{ extern int infd;
fprintf(stderr,"Not enough memory for %s\n",what);
close(infd);
fprintf(stderr,"Word Table is full. TYPOS  stopped.\n");
exit(1);
}
```

BIT BUCKET

41 Pebworth Road,
Harrow,
Middlesex,
England. HA1 3UD.

Dear Don,

We U.K. 68 micro users are indebted to magizines such as yours! for it appears that the U.K. micro. mags. have never heard about our 68 processors, and as such are a dead loss to all of us that use them.

Enough of my pleas, now here are a couple of hints for S.W.T.P. 69 A/K owners. As purchased the serial port card MP-S2 comes only half populated, making only one RS232 port available. The following is how to bring the other port into use, and checks that it works.

After reading the appropriate pages on the constructional details under the MP-S2 section in the machine manual, add all the missing components accept IC.9 . Add the DB25 connector then carefully cut the identification plate for this to fit through, reassemble the card initially straping the Rx & TX clocks the same as port B then after carefull checking of your work re-install the card in the machine.

Before we can use port A however the ACIA (MC 6850) has to be initialized, for this I have assumed the card to be in slot 0 and the motherboard is strapped to decode it at E000,and that a terminal is connected to port B. The following listing is all that is required.

```
        7F  DFE1   CLR.   Port vector address + 1
        BD  FDF1   JSR.   ACIAIZ     taken from Allen Clarke
                                     source listings June 80
                                     68 Micro.
        6E 9F F800 JMP    To Monitor.
```

To check port A we have to change the port address in the vector table, to do this we have to load port A address into it, I.E Address DFE0-DFE1 is changed from E004 to E008 on storing this new data in the table all control from the termin-al will be lost. Disconnect the terminal from port B and plug it : to port A do a CR and normal service should be resumed. If not check all work making sure that the CTS line is pulled high. One last point when using port A in the above mode a RESET will all-ways re-write the port vector address to look at port B.

Hint No. 2 .. A correction to Kunihiro Mitadera article extra 4K space on MP-09 card, Dec. 80, 68 Micro.

S.W.T.P. on their 6809 motherboards, both in the 69 A/K chassis (MP-B1) and the heavier S/09 unit (MP-ID) do not fully decode the I/O port addresses. In these units address A8 line is not used, so if using the above mod. anything addressed with in the E1XX address range will put both the ports available, and any chip inserted in IC1. socket on the bus at the same time. In this I assume that the ports are strapped to a base address off E000.

To overcome this on the MP-B3 motherboard we can use one of the spare gates in IC2. Cut the track of A9 leading to pin 4 (IC2.) close to the base of the chip, then re-wire A9 to pin 2 of IC2. connect A8 to pin 1 (IC 2), and link pins 3 & 4 (IC2) together. See fig. 1.

The MP-ID motherboard on the other hand does not have any spare OR gates, and therefore an other SN74LS32 chip is required. This chip can be physically mounted on top of IC11, and known as IC11A, and could be anchored by the supply pins 7&14 being soldered to same pins on IC11. folding all other pins carefully out of the way of anything else. Cut the track of A9 topin 2 (IC11), re-wire A9 to pin 2 (IC11A), connect A8 to pin 1 (IC11A), and then connect pin 3 (IC11A) to pin 2 (IC11).

These mods. will now give full decoding for the I/O ports on the above cards.

Sincerely,

Jim Anderson.

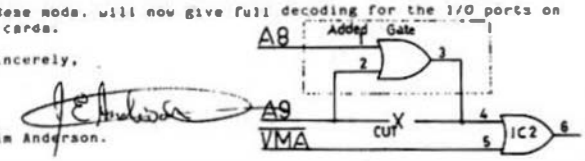


FIG 1

Meteorology Dept.
University of Utah
Salt Lake City, Utah
June 30, 1982

COLOR User Notes
C/O Robert L. May
5900 Cassandra Smith Rd.
Hixson, TN 37343

Dear Robert:

I would like to describe two modifications I have made to my Color Computer and also how to access the 1.8 MHz mode of the 6883 SAM chip.

MODIFICATIONS

LED Pilot Light

The first time I worked with my Color Computer, I inadvertently left it on all night. To remedy this problem I made a LED pilot light using Radio Shack components (LED and housing). The light is powered by tying into TP 12, the +5 volt test point located in the lower left side of the shielded area of the p.c. board, and grounding by connecting to the wide trace around the lower edge of the board (see diagram). The LED is in series with a 220 to 330 ohm current-limiting resistor. It and the housing is mounted in the top plastic cover just to the left of the plastic name label.

1.8 MHz TRS-80 Color Computer

The 6883 SAM chip in the TRS-80 Color computer permits three modes of operation: (1) the normal 0.9 MHz mode; (2) 0.9/1.8 MHz, the well known "POKE 65495,0" mode; and (3) the 1.8 MHZ mode, which will be discussed here. For background, refer to the article, "What Inside Radio Shack's Color Computer?", by Ahrens, Browne, and Scales in the March 1981 issue of BYTE.

The problem with using the 1.8 MHz mode is not one of entering, but rather one of successfully exiting, since the video is out of sync with the microprocessor. The trick, once the microprocessor is in the 0.9 MHz mode, is to resynchronize the 6847 VDG with the microprocessor using the SYNC instruction in a simple machine code subroutine.

To enter the 1.8 MHz mode, simply execute

```
POKE 65497,0
```

To exit the 1.8 MHz mode, execute the subroutine

```
87 FF08    STA $FF08     Exit the 1.8 MHz mode
13         SYNC          Synchronize 6847 with 6809E
39         RTS           Return to BASIC
```

The subroutine may be loaded and called from a BASIC program by adding the following statements (for a 32K machine):

```
1  CLEAR 760,32000              'Protect M.L. Program
2  DATA 183,255,216,19,57       'M.L. Program
3  FOR I = 1 TO 5               'Load M.L. Program
   :READ A
   :POKE 32761+I,A
   :NEXT
4  DEF USR0=32762               'Define M.L. Program Address
      .
      .
      .
   POKE 65497,0                 'Enter 1.8 MHz mode
      .
      .
      .
   X=USR0(0)                    'Exit 1.8 MHz mode
      .
      .
      .
```

In the 1.8 MHz mode the display is meaningless, but anything written to it will appear on returning to the 0.9 MHz mode. My keyboard responds properly in the 1.8 MHz mode, with the exceptions of the letters "G" and "O".

Both the cassette and printer may be used if it is kept in mind that the effective baud are double those of the 0.9 MHz mode. For example, the default printer rate is 1200 baud instead of 600. Likewise, the effective baud rate for the cassette is 3000 in the 1.8 MHz mode as opposed to 1500 baud in the 0.9 MHz mode.

Some basic rules to remember when using the 1.8 MHz mode are:

(1) Debug the program in the slower modes before running in the 1.8 Mhz mode.

(2) Halve the desired printer baud rate before running.

(3) Make a cassette backup of the program BEFORE trying the 1.8 MHz mode.

(4) If the program bombs while in the 1.8 MHz mode. Try typing X=USR0(0). If this fails to exit into the 0.9 MHz mode, hit the RESET button. Sometimes the program is lost or the system hangs up when the system is RESET, thus the cassette backup.

In conclusion, the characteristics of the three modes are:

ITEM	MODE		
	0.9 MHz	0.9/1.8 MHz	1.8 MHz
SPEED	1.0	1.3	2.0
Video	Yes	Yes	No *
Cassette	Yes (1500 b.)	No	Yes (3000 b.)
Printer	Yes	No	Yes **
SOUND	Yes	Yes	Yes
Paddles	Yes	Yes	Yes
Graphics	Yes	Yes	No *

* - Output viewed upon exiting 1.8 MHz mode

** - Baud rate doubled

COMPUTER SYSTEMS CENTER
OF ST. LOUIS, INC.
13441 OLIVE BLVD.
CHESTERFIELD, MO. 63017
(314) 576-6020

August 23, 1982

Dear dealer,

DYNAMITE+ is now available for UniFLEX!

DYNAMITE, our 6809/6800 disassembler for FLEX, has earned compliment after compliment from hundreds of satisfied users. Now UniFLEX users have this indispensable tool available for their system. DYNAMITE+, an enhanced version of the original product, takes 6809 (or 6800) object code and converts it back into assembly language. Then you can inspect the program to see how it works, learn some of the techniques used by expert programmers, and perhaps modify the program to better suit your needs.

The listing on the opposite page was output directly from DYNAMITE+. No editing of any kind was done to produce this superbly readable disassembly. Notice how well DYNAMITE+ handles UniFLEX system calls!

DYNAMITE+ is available to dealers at attractive discounts. Pricing is determined by quantity per order. FLEX and UniFLEX versions may not be mixed to obtain greater discount, but FLEX diskette size (5" or 8" - please specify) may be mixed to achieve desired quantity:

	FLEX	UniFLEX	
Single copy	$ 150.00	$ 300.00	postpaid in U. S.
2-4 copies	$ 75.00	$ 210.00	(30% discount) plus shipping
5 or more copies	$ 60.00	$ 180.00	(40% discount) plus shipping

UniFLEX prices include first year's maintenance.

Terms are company check with order, MasterCard, VISA, or UPS C.O.D. If you prepay, please allow $5.00 per copy for shipping.

DYNAMITE+ is in stock for immediate shipment.

Please let me know if you have any questions about DYNAMITE+. Better yet, call or send in your order today!

Sincerely,

H. Joseph Turner, Jr.
President

Robert E Alexander
82 Miller Rd
Newton, MA 02159
27-Aug-82

Dear Dan,

Everyone is upgrading now to 16-bit microprocessors. IBM has created an industry around the 8088, Tandy has introduced its 68000 computer, and Apple is supposedly working on Lisa, a 68000 machine (which, I suspect, they are having many troubles with). This would explain why they haven't introduced it yet. Remember the Apple III fiasco ? The question comes up: what is the S8-50 industry going to do?

The S6-50 is an 8 bit bus, which effectively puts 16-bit processors off limits. Hazelwood has gotten around this by adding 14 pins and then making a 68000 CPU card. This is an interesting solution, but no one else seems to be doing it. SWTP, Cromix, and DSB all have money invested in 50-pin boxes and boards (as do their customers) so adding pins is not very attractive.

There are, of course, versions of 16 bit processors with 8 bit data busses. The 8088 is an 8 bit version of the 8086 and the 68008 is an 8 bit version of the 68000. I speculate that SWTP, et al are planning to put the 68000 on the S5-50 bus, the assumption being that we should stick with Motorola chips. I feel this is a bad move and could be disastrous for the industry.

The 68008 is in no way software-compatible with the 6800 or 6809. Its bus timing can be made compatible, but doing so slows down the processor. Furthermore, a 68008 system would put our industry in direct competition with the likes of Tandy and Apple. Up till now, we've offered something different: the 6800 was better than the 8080, the 6809 was better than the Z80, the S5-50 was better defined than the S-100, FLEX was better than CP/M, and OS-9 was better than MS-DOS. If we use a 68008, we will be selling systems that use the same instruction set as Tandy and Apple machines, but it will run slower!

Our only alternative, as I see it, is to do something different. National Semiconductor has come out with its 16000 processors, including an 8 bit version, the 16008. It has a cleaner, more powerful instruction set than the 68000. Its advantages:

1) Eight registers that are truly general purpose, not divided into data and address registers which behave differently.

2) Special purpose registers including two stack pointers, a frame pointer for accessing local variables, a static base pointer for accessing global variables, and a register that points to tables used for linking to external routines (writing a linker for the 16000 is trivial!). The 16000 is an excellent Pascal machine.

3) Powerful addressing modes such as array addressing in which you supply the base address of the array, the index of the element you want, and the length of each entry (1, 2, 4 or 8 bytes). The index is multiplied by the length and added to the base to get the effective address. Another mechanism makes addressing records a snap. Unlike the 68000, instructions may have both operands in memory. For example, the 16000 instruction MULT 10$,10$ uses the top-of-stack addressing mode for each operand. Both operands are popped from the stack and multiplied together. The result is placed on the stack. The 16000 is an excellent TURIN machine.

4) The 16000 has string handling instructions, a computed jump (for implementing CASE statements), and bit field extraction that doesn't depend on byte or word boundaries.

What are its disadvantages? Its registers are all 16 bits instead of the 32 bits on the 68000, it only addresses 64K, and its bus is not 6800 compatible. Even so, I believe it's a better chip than the 68000.

The bus problem can be solved easily. Put the 16008 on its own card with its own memory. Another processor on the S5-50 bus, either a 6800 or 6809, should be able to access a small part of the 16008's memory, thus allowing communication between processors. The 6800/9 will handle all the I/O, leaving the 16000 free for more weighty tasks. The 16000 no longer has to interface to our 6800 compatible busses. In fact, all of us with old systems can upgrade to a multiprocessor system by adding one board. (With a setup like this, we could even use the 16016 or 16032 which are faster and use a 16 bit data bus.)

The 16 bit registers should not cause severe problems. The 16008's power in other areas should make up for it. The 64K limit is the only big problem. Perhaps a memory-swapping scheme controlled by a 6809 accessing 2Mbytes of memory could alleviate the problem.

We've been loyal Motorola users for years. But now, with 68000 based systems all over the place, it's time to be different again. We will have to rewrite our software anyway, we may as well use the best hardware available.

Sincerely,

Robert E Alexander

```
                        * disassembly by dynamite+ of fl

                                info    Filter to block 'L's
                                info    (lo, after assembly)
                                info    from clearing screen.
                                info    by Scott Schaefer!e
                                info    June 28,1982

                        * system name equates

                0005    term    equ     5
                000C    read    equ     12
                000D    write   equ     13

                        * oranda!d pre-named label equates

                0040    buffer  equ     $0040

0000 CC   0000          start   ldd     #$0000
0003 113F 0C                    sys     read,buffer,$0081
0006 1025 0021                  lbcs    error
000A 1083 0000                  cmpd    #$0000
000E 26   03                    bne     ok
0010 113F 05                    sys     term
0013 86   40            ok      lda     >buffer
0015 81   CC                    cmpa    #$cc
0017 26   05                    bne     output
0019 C6   07                    ldb     #7
001B D720 07            output  stb     >buffer

001D CC   0001          output  ldd     #$0001
0020 113F 0D                    sys     write,buffer,$0081
0023 24   01                    bcc     start

0025 34   06            error   pshs    a,b
0027 CC   0001                  ldd     #$0001
002A 113F 0D                    sys     write,errmsg,$0004
002D 35   06                    puls    a,b
002F 113F 05                    sys     term

0032                            org     $0041

0041 45 72 72 6F        errmsg  fcc     "Error!"

                        end     start
```

27 September, 1982

Mr. Don Williams Sr.
68 Micro Journal
Hixson, TN. 37343

Dear Don,

In the process of implementing the General Version of Flex9, there was a requirement to commit the boot program to firmware. Since my system consisted primarily of CREATIVE MICRO SYSTEMS hardware, I purchased the 9617 EPROM PROGRAMMING HEAD. The unit is self contained and does not require any external support. A 50 pin connector interfaces with the PIA I/O socket on the Micro Processor Board. The controlling software is supplied on 2716 devices and is available for the 6800, and the 6502 as well as the 6809. The Programmer will work on single voltage Eproms such as the 16K 2716, or the 32K 2732, family of devices. The 9617 will operate with other vendors hardware. The requirements are, 2 PIA Ports and a 5 volt source. At the present time, there are two versions of firmware available for the 9617. One follows the format of the popular DEBUG09 (MICROWARE), the other follows the format of the in house SYSMON09, which I preferred since it was easier to enter data to memory from the Terminal. The Source for the program is included so it should be relatively easy to adapt the program to your own requirements.

Duplicating or Burning a new EPROM is an extremely simple task if you are using the 9617. For example: With the firmware plugged in at $E800, the following procedures apply: Enter P from the keyboard. The Banner appears, followed by a CR and LF and the system prompt (Pe). Commands are entered subsequent to the prompt which indicates that the system is ready to process a command.

Enter the Command TMS2516 CR.

This first command sets up the Parameters for 16K devices such as the 2716 Eprom. Enter TMS2532 for 32K devices, ie: 2732. Commands are entered by depressing the CR key. Errors occuring prior to CR entry can be corrected with a Control H (Backspace). The next step depends on what you want to do. If you wish to duplicate a PROM place the Master in the Zero Force socket.

Enter the command READ ssss,eeee,rrrr,pppp CR.

    ssss=Start Resident address in Eprom
    eeee=End   "   "   "
    rrrr=Ram address of program to be burned
    pppp=Prom-Offset, normally $0000

The ram address is the address where the program contained in the Master will be stored.

Next Enter the command VERF ssss,eeee,rrrr,pppp CR.

This step verifies that the Master program was copied to memory correctly. If you wish to change something in the master, ie, addresses or the program itself, then:

Enter EXAM HHHH CR.

In this case HHHH equals the location in memory where the change(s) is required. Entering a change or entering a space (space bar) will advance the address location. Entering the ^ key will decrement the address. The next step is to remove the master, and insert the intended duplicate.

Enter TEST sss,eeee,rrrr,pppp CR.

If the Eprom inserted is fully erased, then the terminal will display the system prompt, if not, the addresses of non-clear areas will be reported.

Enter PROG ssss,eeee,rrrr,pppp CR.

The Programming LED on the 9617 will remain "ON" until the programming process is complete. The last step is:

Enter VERF ssss,eeee,rrrr,pppp CR.

This step will compare the contents of the prom against the program stored in memory. If there were no errors, which is the usual case, the Pe prompt will indicate termination of the process and the duplicate may be removed from the 9617. There are 4 additional commands available to the user. They are VIEW,STAT,CLR, and CRCC which will be of interest to owners of the CREATIVE MICRO SYSTEMS 9617 EPROM PROGRAMMING HEAD.

For those of you who intend to implement the FLEX09 DOS on CMS hardware, make the following additions to the duplicate Prom firmware. I used the space available on the Programing Prom, address $E000, for the Boot program. It was necessary to use a duplicate Eprom since in its original form, an End of Command table blocked usage of the remaining space. The following listing applies to in in house SYSMON Firmware:

```
$EDD0...46 4C 45 45        NEW COMMAND "FLEX"
 EDDF...00                 END OF STRING
 EDE0...EDE4               VECTOR TO FLEX BOOT
 EDE2...00 00              END COMMAND TABLE
 EDE4...86 E010            START OF BOOT
```

ENTER THE BALANCE OF BOOT, page 86, Appendix G, Adaptation Guide.

One final word, the documentation is excellent and the simplicity of its use makes the unit a functional addition to one's Computer System. From the user viewpoint, I find it very comfortable and easy to use.

I give the CMS EPROM PROGRAMMING HEAD a rating of AAA.

Anthony J. Gasbarro
23 Centre Street
E.Sullivan, N.H. 03445
603 847-9797

```
P OCEDURE MODEM
0000      (* THIS PROCEDURE PERMITS THE COMPUTER TO ACT AS A TERMINAL *)
003E      (* VIA THE /M1 SERIAL PORT MODULE DEVICE DESCRIPTOR.  THE   *)
007C      (* "}" (125) CHAR TURNS THE PRINTER ON OR OFF.  THE "|"     *)
00BA      (* (124) CHAR TURNS A DISKFILE BUFFER ON OR OFF.  THE "{"   *)
00F8      (* (123) CHAR TRANSMITS A LINE FEED ONLY.  THE "~" (126)    *)
0136      (* CHAR EXITS OR ENDS THE PROGRAM.  THE MODEM PORT SHOULD   *)
0174      (* HAVE ALL ITS ATTRIBUTES TURNED OFF (IE: ECHO LF, ETC.).  *)
01B2      (* THE CRT TERMINAL SHOULD ALSO HAVE ALL ITS ATTRUBUTES OFF.*)
01F0      (* THE "@" (64) CHAR ALLOWS YOU TO TRANSMIT ASCII DISKFILES.*)
022E      (* THIS PROGRAM WAS SET UP USING THE 25TH LINE ON A H-19    *)
026C      (* TERMINAL. YOU MAY HAVE TO CHANGE THE LINES AS NEEDED.    *)
02AA      (* THIS PROGRAM ALSO USES AN EXTERNAL PROCEDURE (GETBYTE)   *)
02E8      (* WHICH IS A MACHINE LANGUAGE PROGRAM FOR INPUTTING CHAR'S *)
0326      (* INTO THE PROGRAM. THE /M1 DEVICE DESCRIPTOR MODULE IS    *)
0364      (* SIMPLY A SERIAL PORT MODULE WITH ALL IT'S OPTIONS TURNED *)
03A2      (* OFF SO THAT IT DOESN'T CAUSE PROBLEMS.                   *)
03E0      (* THE PROGRAM BARELY HAS ENOUGH TIME TO WORK PROPERLY AT   *)
041E      (* 300 BAUD. SO EXERCISE CAUTION WHEN ADDING ANY EXTRA STUFF*)
045C      (* TO IT.                                                   *)
049A      (* I'VE HAD TROUBLE WITH THE I/O MODULES FILTERING STUFF I  *)
04D8      (* WANTED TO GO THROUGH (IE: ESCAPE) THUS THE UNPRETTY.     *)
0516      (* SPECIAL CHARACTERS FOR VARIOUS FUNCTIONS. IF YOU ARE MORE*)
0554      (* INGENIOUS THAN I AM PLEASE LET ME KNOW WHAT SECRETS ALLOW*)
0592      (* ESC'S TO GET THROUGH PROPERLY. ALSO WHEN I USED THE OS-9 *)
05D0      (* USER'S GROUP SBB. I EXPERIENCED SOFTWARE CRASHES. I'VE   *)
060E      (* FILTERED OUT MOST OF THE XON/XOFF ETC. STRANGE STUFF, BUT*)
064C      (* SOMETHING CAN STILL CAUSE A CRASH. IF ANYONE FIGURES IT  *)
068A      (* OUT, LET ME KNOW TOO. THIS PROGRAM WORKS WELL HERE IN    *)
06C8      (* ARIZONA, AND ALL THE SBB'S I'VE TRIED COOPERATE VERY WELL*)
0706      (* BY E.W.BOLLINGER ON AUG. 16, 1982.       VER 1.0         *)
0744      (* LAST MODIFIED ON SEPT 12, 1982 BY E.W.BOLLINGER          *)
0782      (* E.W.BOLLINGER 10810 N 91ST AVE, PEORIA,ARIZONA 85345     *)
07C0      (* PHONE NUMBER: 602-979-4451.                              *)
07FE      (* THIS PROGRAM IS HEREBY RELEASED INTO THE PUBLIC DOMAIN   *)
083C      (* FOR THE FREE AND PRIVATE USE OF ANYONE WHO WANTS TO USE  *)
087A      (* IT FOR THEIR PRIVATE PURPOSES.                           *)
08B8      DIM PORT,DISKFILE,PRINTER,COMMAND,XMTFILE:STRING
08CF      DIM PATH,PPATH,FPATH,A,LF,XPATH,CHAR:BYTE
08EE      DIM F,P,X,ERRNUM:INTEGER
0901      ON ERROR GOTO 100
0907      PORT:="/M1"
0911      PRINTER:="/P1"
091B      F:=0
0922      P:=0
0929      X:=0
0930      LF:=10
0937      COMMAND:="TMODE -ECHO -LF -BSB -BSL -PAUSE"
095E      SHELL COMMAND
0963      OPEN #A,PORT:UPDATE
096F      PRINT CHR$(27); "j"; \(* save cursor position *)
0993      PRINT CHR$(27); "x1"; \(* enable 25th line *)
09B4      PRINT CHR$(27); "Y8 "; \(* move cursor to 25th line *)
09DE      PRINT CHR$(27); "p"; \(* enter reverse video *)
0A01      PRINT "Modem Sw'n! ON/OFF }=Printer |=Diskfile {=Line Feed ~=Exit
@=Xmtfile"
0A4C      PRINT CHR$(27); "q"; \(* Exit reverse video *)
0A6E      PRINT CHR$(27); "k"; \(* return to saved cursor position *)
0A9D      (* main body of the modem program *)
0AC1      LOOP
0AC3 10     PATH:=A
0ACE        RUN GETBYTE(CHAR,PATH)
0AE0        IF CHAR<>0 THEN
0AE9          PRINT CHR$(CHAR);
0AF0          IF P=1 THEN
0AFC            IF CHAR<>10 THEN
0B08              PRINT #PPATH,CHR$(CHAR);
0B14            ENDIF
0B16          ENDIF
0B18          IF F=1 THEN
0B24            IF CHAR<>10 THEN
0B30              PRINT #FPATH,CHR$(CHAR);
0B3C            ENDIF
0B3E          ENDIF
0B40        ENDIF
0B42        PATH:=0
0B49        RUN GETBYTE(CHAR,PATH)
0B58      EXITIF CHAR=126 THEN
0B64      ENDEXIT
0B68        IF CHAR=64 THEN
0B74          PRINT CHR$(10); CHR$(13);
0B7E          XMTFILE:=""
0B85          PRINT "ENTER FILENAME: ";
0B9A          LOOP
0B9C            GET #0,CHAR
0BA5            PRINT CHR$(CHAR);
0BAC          EXITIF CHAR=13 THEN
0BB8          ENDEXIT
0BBC            XMTFILE:=XMTFILE+CHR$(CHAR)
0BC9          ENDLOOP
0BCD          PRINT CHR$(10); CHR$(13);
0BD7          X:=1
0BDE          OPEN #XPATH,XMTFILE:READ
0BEA          SEEK #XPATH,0
0BF3          WHILE EOF(#XPATH)=FALSE DO
0BFF            GET #XPATH,CHAR
0C09            PUT #A,CHAR
0C13            IF CHAR=13 THEN
0C1F              PUT #A,LF
0C29            ENDIF
0C2B            PRINT CHR$(CHAR);

0C32            IF CHAR=13 THEN
0C3E              PRINT CHR$(LF)
0C44            ENDIF
0C46          ENDWHILE
0C4A          CLOSE #XPATH
0C50          X:=0
0C57          PRINT CHR$(10); CHR$(13);
0C61          PRINT "TRANSMISSION OF "; XMTFILE; " IS DONE."
0C85          PRINT CHR$(10); CHR$(13);
0C8F        ENDIF
0C91        IF CHAR=125 THEN
0C9D          IF P=0 THEN
0CA9            OPEN #PPATH,PRINTER:WRITE
0CB5            P:=1
0CBC            GOTO 50
0CC0          ENDIF
0CC2          IF P=1 THEN
0CCE            CLOSE #PPATH
0CD4            P:=0
0CDB            GOTO 50
0CDF          ENDIF
```

```
0CE1           IF
0CE3        IF CHAR=124 THEN
0CEF          IF F=0 THEN
0CF8            PRINT CHR$(10); CHR$(13);
0D05            PRINT "ENTER FILENAME: ";
  1A            DISKFILE:=""
0D21            LOOP
0D23              GET #0,CHAR
0D2C              PRINT CHR$(CHAR);
0D33            EXITIF CHAR=13 THEN
0D3F            ENDEXIT
0D43              DISKFILE:=DISKFILE+CHR$(CHAR)
0D50            ENDLOOP
0D5A            PRINT CHR$(10); CHR$(13);
0D5E            CREATE #FPATH,DISKFILE:WRITE
0D6A            F:=1
0D71            GOTO 50
0D75          ENDIF
0D77          IF F=1 THEN
0D83            CLOSE #FPATH
0D89            F:=0
0 90            GOTO 50
0D94          ENDIF
0D96        ENDIF
0D98        IF CHAR=123 THEN
0DA4          PUT #A,LF
0DAE          GOTO 50
0DB2        ENDIF
0DB4        IF CHAR<>0 THEN

0DC8          PRINT #A,CHR$(CHAR);
0DCC        ENDIF
0DCE 50   ENDLOOP
0DD5      COMMAND:="TMODE ECHO LF BSB BSL PAUSE"
0DF7      SHELL COMMAND
0DFC      IF P=1 THEN
0E08        CLOSE #PPATH
0E0E      ENDIF
0E10      IF F=1 THEN
0E1C        CLOSE #FPATH
0E22      ENDIF
0E24      IF X=1 THEN
0E30        CLOSE #XPATH
0E36      ENDIF
0E38      CLOSE #A
0E3E      PRINT CHR$(27); "y1" \(* DISABLE 25TH LINE *)
0E5F      END
0E61      (* ERROR TRAPS SECTION *)
0E7A 100  ERRNUM:=ERR
0EB3      PRINT "ERROR #: "; ERRNUM
0E94      PRINT CHR$(10); CHR$(13);
0E9E      IF ERRNUM=216 THEN
0EAA        IF X=1 THEN
0EB6          X:=0
0EB0          GOTO 10
0EC1        ENDIF
0EC3      ENDIF
0EC5      IF ERRNUM=215 THEN
0ED1        IF X=1 THEN
0EDD          X:=0
0EE4          GOTO 10
0EE8        ENDIF
0EEA      ENDIF
0EEC      IF ERRNUM=214 THEN
0EF8        IF X=1 THEN
0F04          X:=0
0F08          GOTO 10
0F0F        ENDIF
0F11      ENDIF
0F13      IF P=1 THEN
0F1F        CLOSE #PPATH
0F25      ENDIF
0F27      IF F=1 THEN
0F33        CLOSE #FPATH
0F39      ENDIF
0F3B      IF X=1 THEN
0F47        CLOSE #XPATH
0F4D      ENDIF
0F4F      CLOSE #A
0F55      COMMAND:="TMODE ECHO LF BSB BSL PAUSE"

0F77      SHELL COMMAND
0F7C      PRINT CHR$(27); "y1"; \(* DISABLE 25TH LINE *)
0F9E      END
```

```
                 **********************************
                 *
                 *  Basic09 GetByte Procedure
                 *     called by: RUN GETBYTE(byte,pathbyte)
                 * Accepts a character byte from path given in pathbyte,
                 * if data is available, otherwise it returns a null
                 * ( 00 ) byte.
                 *
                 *     This procedure is used primarily by the Basic09
                 * procedure called MODEM. Getbyte tries to filter out
                 * most of the garbage that can cause problems. But
                 * sometimes things can still get thrown.
                 *
                 * by  E.W.Bollinger on Aug. 6, 1982   ver 1.0
                 *     last modified on Sept 12,1982.
                 **********************************
                          nam   Getbyte
                          use   /D0/DEFS/OS9defs
                          opt   l
                          ifp1
                          endc
0021             TYPE     set   SBRTN+OBJCT
0081             Revs     set   Reent+1
0000 87CD00A9             mod   Getend,Getnam,Type,Revs,Getent,0
000D 47657462    Getnam   fcs   /Getbyte/
0014 EC62        Getent   ldd   2,S              get parameter count
0016 10270084             lbeq  Getxit        quit if no parameters
001A AE68                 ldx   8,S              addr of 2nd parameter
001C A684                 lda   0,X              get path byte
001E AE64                 ldx   4,S              addr of 1st parameter
0020 6F84                 clr   0,X              initialize to null(0) byte
0022 C601                 ldb   #SS.RDY
0024 103F8D               OS9   I$GSTT        is data ready?
0027 2575                 bcs   Getxit        not then return
0029 108E0001             ldy   #1
002D 103F89               OS9   I$READ        read a byte
0030 A684                 lda   0,X
0032 847F                 anda  #$7F             strip parity
0034 8101                 cmpa  #$01             test for bad data char's
0036 2768                 beq   Fix                if bad then fix them
0038 8102                 cmpa  #$02
003A 2764                 beq   Fix
003C 8103                 cmpa  #$03
003E 2760                 beq   Fix
0040 8104                 cmpa  #$04
0042 273C                 beq   Fix
0044 8105                 cmpa  #$05
0046 2738                 beq   Fix
0048 8106                 cmpa  #$06
004A 2734                 beq   Fix
004C 810B                 cmpa  #$0B
004E 2730                 beq   Fix
0050 810C                 cmpa  #$0C
0052 274C                 beq   Fix
0054 810E                 cmpa  #$0E
0056 2748                 beq   Fix
0058 810F                 cmpa  #$0F
005A 2744                 beq   Fix
005C 8110                 cmpa  #$10
005E 2740                 beq   Fix
0060 8111                 cmpa  #$11
0062 273C                 beq   Fix
0064 8112                 cmpa  #$12
0066 2738                 beq   Fix
0068 8113                 cmpa  #$13
006A 2734                 beq   Fix
006C 8114                 cmpa  #$14
006E 2730                 beq   Fix
0070 8115                 cmpa  #$15
0072 272C                 beq   Fix
0074 8116                 cmpa  #$16
0076 2728                 beq   Fix
007  8117                 cmpa  #$17
007A 2724                 beq   Fix
007C 8118                 cmpa  #$18
007E 2720                 beq   Fix
0080 8119                 cmpa  #$19
0082 271C                 beq   Fix
0084 811A                 cmpa  #$1A
0086 2718                 beq   Fix
0088 811B                 cmpa  #$1B
008A 2714                 beq   Fix
008C 811C                 cmpa  #$1C
008E 2710                 beq   Fix
0090 811D                 cmpa  #$1D
0092 270C                 beq   Fix
0094 811E                 cmpa  #$1E
0096 2708                 beq   Fix
0098 811F                 cmpa  #$1F
009A 2704                 beq   Fix
009C A784                 sta   0,X
009E 5F          Getxit   clrb
009F 39                   rts
00A0 8B20        Fix      adda  #$20
00A2 A784                 sta   0,X
00A4 20F8                 bra   Getxit              convert to a more suitable
00A6 CF5057               emod
00A9             Getend   equ   *
                          end

00000 error(s)
00000 warning(s)
$00A9 00169 program bytes generated
$01D1 00465 data bytes allocated
$137B 04987 bytes used for symbols
```

DIET-TRAC, A Review

Upon setting up and "Running" the Program as instructed I found that all Programs ran as the advertising literature and Particularily as the "READ-ME.TXT" described they would. I found all associated Programs to be very "friendly" and I suspect that the most novice operator, once the Program was loaded, could successfully follow the Prompting and Produce valid results.

As I used the Program, skimming through each of it's sections, I was Pleased with each sections aPPlicability to an individualized weight control Program. The "file" section seemed to be to only section which Provided any ackwardness to the ParticiPent and this only from the standPoint that each client needed to keeP such detailed records of their eating Patterns. I'm not really sure that I can criticize this entirely since it adds decliPine to the dieters regime, it just seems that it might be simPlified a bit but I'll sPeak to this later.

I am Particularily Pleased with the information which the "Plan" section comPiles and how it shows the client, for instance, his or her Present calorie consumPtion comPared to what it would be after the client reached their goal and what it should be while they are dieting to reach that goal. I was also imPressed that enough foresight was given to allow a diet to be Planned by C/P/F rather than strictly by calories. This is an imPortant consideration esPecially when theraPutic diets for reasons other than weight loss need to be Planned.

As I come to the exchange list I would like to say that I have found, in my clinical situation, that Patients are often confused by 1/2 exchange rates. for examPle 6-1/2 breads or 2-1/2 vegetables. Please do not misunderstand that I consider the 1/2 exchange PrinciPle to be invalid. I do not. It is more valid than using whole numbers. The Problem lies in comPlience of the Patient and I have found that the simPler the diet is to understand the more comPlient the Patient will be. There is another Point in this area that I must take a stronger excePtion to and that concerns the number of grams of Protein you allow in the vegetable exchange. In your calculations you use 5 grams of carb Plus 1 gram of Protein. I have always understood that there were 2 grams of Protein in one vegetable exchange which would add an additional 4 calories to each vegetable exchange. Other than this I find no areas in your theory that I would fault. I must admit that I am not familiar with the formula you use to determine Basal Metabolic Rate but it certainly seems to work in all cases I have tried. In my clinical situation we use testing to determine BMR.

Concerning the "calendar" I have the following suggestion. It has been my clinical experience that daily weigh-ins can become discouraging to the client and I never encourage the patient to weigh his or herself more than once a week. With this in mind it would be my suggestion that it be printed at weekly rather than daily intervals. There is a Program I would like to see added to this one which might make it just that much more useful to the client and therefore to the dietitian.

As I mentioned earlier the simplicity of instructions and the simplicity of following them are of Paramount importance in the all important area of Patient compliance. If the clients did not need to sit down each day or week and figure out, via longhand, a diet based on the exchange list they might be more likely to stick to that plan and achieve their goal. It seems to me that this is what they are looking for when they seek out the "fad" diets (that is besides quick weight loss promises). If you could come up with a data based food exchange list from which the client could select favorite foods and then have those selected foods become a Part of the clients file it would then be possible to generate a random cyclical menu that the client could use daily. In addition to this the menu could show, by each day, the Veg., Bread, Meat, Milk, Fruit, and Fat consumed. This would make it easier to keep records for the file.

All-in-all I must say that I find this to be a helpful dietetic aid and one that may well Prove to be a useful "gimic" in an age when it seems we need gimics to get and keep Persons interested in caring for themselves.

Sincerely yours,

Mary J. Benson R.D.

Don Williams
'68' Micro Journal
3900 Cassandra Smith Road
Hixson,
TN 37343
U.S.A.

Ted Edwards
The Old Forge,
Little St. Mary's,
Long Melford,
Suffolk CO10 9HX.
England

Dear Don, 12th September 1982

First I would like to say how much I enjoy reading your magazine. It has about the right balance of articles. Keep on going just as you are!

Here are a couple of things which may interest some of your readers. Flex for the 6809, when it boots, auto-configures for a lot of the hardware attributes of the system it is being run on. Now, on my system it gets many of these things wrong and I had to put an SBOX command string in my STARTUP file. What with one thing and another (reserving printer space, setting TTYSET parameters etc.), STARTUP was taking longer and longer to run at boot time. Anyway I found that Flex uses the byte at CC33 for a system configuration map as shown in the table. I used 'FIX' to modify Flex at C8E3 where the auto-configure starts and loaded CC33 with my own configuration by putting a short patch at C8E3. Whilst many people would rather use SBOX it is still worth knowing about this feature.

LOCATION CC33 BIT SIGNIFICANCE :-

	0	1	
BIT 0	NO	YES	EXTENDED ADDRESSING ?
BIT 1	ABSENT	PRESENT	INTEVAL TIMER ?
BIT 2	16 SLOT	4 SLOT	PORT SPACING ?
BIT 3	ABSENT	PRESENT	REAL TIME CLOCK ?
BIT 4	NO	YES	ON-BOARD CPU RAM ?
BIT 5	60 Hz	50 Hz	POWER LINE FREQUENCY ?
BIT 6	NO	YES	CLOCK STRETCH ?
BIT 7	1 MHz	2 MHz	CPU CLOCK RATE ?

Modification to Flex 2.0:3 (Addresses may be different in other versions):-

ADDRESS	ORIGINAL	NEW
C8E3	F6 E005	86 24 B6
C8E6	27 15	CC33
C8E8	C1 FF	7E C000

This mod stops Flex auto-config. The new configuration is at C8E4 and in the above example is set to $24 ie. 1 MHz,no clock stretch,50Hz,no CPU ram, no RTC,4 slot,no timer,not extended addressing.

The other item is a short command designed to be run in STARTUP that 'plugs in' a memory command extension table to Flex. It is designed to be universal and is better than just overlaying with a GET command as MEMEND can vary for a host of reasons. The two commands actually shown in this listing are for my system and so must be changed to suit whatever memory based commands are in your system. I have a 2K 'firmware' ram at $E800 (soon to be $F000) that contains my memory resident commands. This is a very fast way of executing often used utilities.

Best wishes,

Ted Edwards

```
                    *This prog is designed to be run in STARTUP
                    *and installs a memory resident command ext-
                    *tension at high end of memory, adusting
                    *MEMEND in Flex accordingly. It contains
                    *within itself the command lookup table
                    *which is easily patched. The program
                    *also patches the command table into Flex.
                    *
                    *
C100                        ORG     $C100
C100 20   12        START   BRA     STAR
C102 01             VN      FCB     $1          version
                    *
                    *This is start of actual command table. For
                    *format description see Flex Advanced
                    *Guide (page 16, Memory Resident Commands)
                    *It is important that the label 'TEND'
                    *should always be associated with the
                    *command table terminator. Given this,
                    *the table can be expanded to cater for any
                    *number of memory resident commands.
C103 44 41 54 45    TABLE   FCC     /DATE/      Command name
C107 00                     FCB     0
C108 E800                   FDB     $E800       Entry address
C10A 54 49 4D 45            FCC     /TIME/      Command name
C10E 00                     FCB     0
C10F E804                   FDB     $E804       Entry address
C111 00             TEND    FCB     0           Table terminator
                    *
                    *EQUATES :-
             C003   WARMS   EQU     $C003       Flex entry
             CC2B   MEMEND  EQU     $CC2B       Mem end adr vector
             CC12   UCTABA  EQU     $CC12       User Command Table
                    *
```

-2-

```
                    *CONSTANTS:-
C112 000E           LEN     FDB     TEND-TABLE Length of table
                    *
C114 FC  CC2B       STAR    LDD     MEMEND      Get old end of mem
C117 B3  C112               SUBD    LEN         Make room for table
C11A FD  CC2B               STD     MEMEND      Store back in Flex
C11D 1E  01                 EXG     D,X
C11F 30  01                 LEAX    1,X         Adjust adr pointer
C121 BF  CC12               STX     UCTABA      Pointer to Flex
C124 10DE C103              LDY     OABLE       Load adr pointer
                    *
C128 A6  A0         LOOP    LDA     0,Y+        Get byte from table
C12A A7  80                 STA     0,X+        Store in memory
C12C 108C C112              CMPY    OEND+1      Finished ?
C130 26  F6                 BNE     LOOP        Branch if not
                    *
C132 7E  C003               JMP     WARMS       EXIT
                            END     START

0 ERROR(S) DETECTED
```

41 Pebworth Road,
Harrow,
Middlesex,
England. HA1 3UD
TEL. 01-422 4724

Dear Don,

Recently, a new 68 Micro special interest user Group has been formed which is affiliated to the United Kingdom A.C.C. Our aim is to bring together as many U.K. and even European users in an exchange of information, meetings, newsletter,and run a library.

To all your readers concerned please send me your name and address together with a S.A.E.

The next request goes out to other 68 user groups within reach of your circulation boundaries in view of setting up an exchange of information through newsletters etc., to help us keep in touch with what is happening throughout the 68 world.

Hoping that you can find space for this in your columns, and please keep up the good work.

Sincerely,

Jim Anderson.

RAIL RUNNER is available from Computerware dealers or directly from Computerware at Box 668, Encinitas, Ca. 92024, (714) 436-3512. It costs $ 21.95on cassette and $26.95 on disk, plus $2.00 for shipping and handling.

Motorola Announces Availability of the MC6822 Industrial Interface Adapter

Austin, Texas, August 10, 1982... Motorola Microprocessor Division announces availability of the MC6822 Industrial Interface Adapter. This device will provide interface capability for the M6800 family NMOS/HMOS MPU's to CMOS and other higher voltage levels directly, without any additional circuitry. Users will have the greatest peripheral interface capability with reduced circuit component count.

The MC6822 offers several features including: open drain peripheral lines capable of interfacing with higher voltage levels (up to 18v), two bidirectional 8-bit buses for interface to peripherals, two programmable control registers, and two programmable data direction registers. In addition there are four individually controlled interrupt input line (two usable as peripheral control outputs), handshake control logic for input and output peripheral operation, program controlled interrupt enable and disable capability, 8-bit bidirectional data bus for MPU interface. The part is pin compatible with the MC6821 (PIA) and is TTL compatible.

Dear Don,

I would like to announce that after a year the spelling checker is now ready. SPELLB contains a 14,000 word common dictionary and a 120,000 word main dictionary. With larger files the checking speed is 1700 words per minute. Other features are a personal word list and the ability to call dictionaries of special words such as Medical, religious or whatever.

A Help command will let you view correctly spelled words similar to the word reported as misspelled. Other options are to Add the word to your personal word list, Delete word, Mark the word for later processing, Replace the word with correction and View the word in context during the Update run.

The program runs under Flex9. It occupies 1000 sectors and is supplied on one 8" disk or three 5" disks. A special introductory price is $125.00.

PALM BEACH SOFTWARE

3648 LANTANA ROAD 305 967-3660

LANTANA FL 33462

COMPUTER SYSTEMS CENTER
OF ST. LOUIS, INC.
13481 OLIVE BLVD.
CHESTERFIELD, MO. 63017
(314) 576-5020

September 23, 1982

Mr. Don Williams, Publisher
68 MICRO JOURNAL
5900 Cassandra Smith Road
Hixson, TN 37343

Dear Don,

We received the October, 1982, issue of 68 MICRO JOURNAL today, and were very happy to see that you printed some very favorable comments about DYNACALC, our Electronic Spread-Sheet program for 6809 computers. Your positive opinion is much appreciated, and is shared by hundreds of DYNACALC users all over the world.

Unfortunately, the DYNACALC commentary on page 12 (part of COLOR User Notes, by Robert L. May), states that there is a relationship between our DYNACALC and VisiCalc, a well-known product available for Apple, Tandy, and IBM personal computers. This is not the case. DYNACALC is our own original product, written in 6809 assembly language for 6809 systems only. While many features of DYNACALC are similar to features found in other well-designed spread-sheet programs, DYNACALC has its own personality, and many commands and functions not found in other products.

For example, the article mentions DYNACALC's sorting ability. To our knowledge, no other spread-sheet program available on any computer system has a built-in sort command. This may seem a small point, but in practice this command opens up whole new application areas to the spread-sheet concept.

On page 13 Mr. May makes a comment to the effect that using DYNACALC takes as much thought and planning as does writing a program in BASIC or Pascal. There is a lot of truth to that, but the idea of programs like DYNACALC is to let that thought and planning be done by non-programmers. The person who sets up the spread-sheet is usually the one who will later use it. This allows the power of the computer to be brought directly to the level of the end user.

DYNACALC is a trademark of Computer Systems Center
VisiCalc is a trademark of VisiCorp, San Jose, California

YOU CAN BUILD ALMOST ANYTHING WITH OUR SOFTWARE

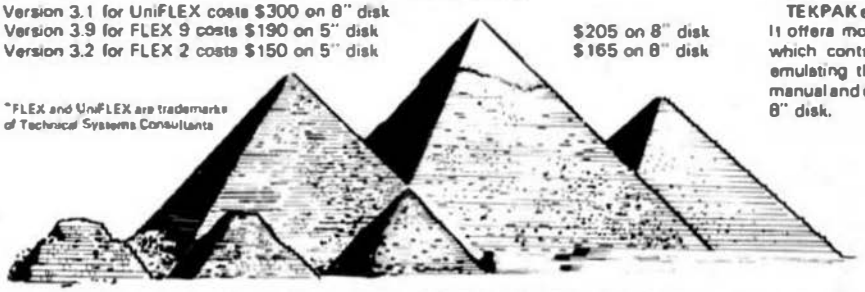
Scientific and Engineering models (full 9 digit precision), Financial Models, Database Managers, Compilers, Graphics packages and even Pyramids!

LUCIDATA PASCAL has been used world-wide for more then 4 years and is the easiest, friendliest and space efficient implementation there is for 6800/6809 based machines. There is no messy assemb r pass — your programs compile directly into our memory efficient p-code which is common to 6800 or 6809 FLEX™ and UniFLEX™ versions. The extremely flexible device I/O and linkage to external routines have earned our Pascal its international reputation as being the best one to use in a whole range of applications.

Continuous development over the years as well as feedback from a very wide customer base, has prompted many major enhancements. Separately compilab OVERLAYS mean there are NO LIMITATIONS to the size of the program you can develop. And our VIRTUAL MEMORY mode still means that even the smal st configuration can run our Pascal System. EXTERNAL functions to perform UCSD-type string handling are included, as well as the ability to define your own to suit your requirements. (The UniFLEX version has additional external functions to make full use of the multi-tasking environment offered by UniFLEX). ASSIGNMENT of external file names from within your program gives you complete control over your files. Auto-restart facility, Scalar I/O, automatic minimisation of memory usage . . . and much, much more. BUT (and this is the most important fact of all) you can write in Lucidata Pascal without writing non-ISO Standard code. This means that Lucidata Pascal programs are totally transportable at the source level between all machines whose compilers adhere to this internationally agreed Standard. (Lucidata Pascal is also available for CP/M and HDOS machines).

Version 3.1 for UniFLEX costs $300 on 8" disk
Version 3.9 for FLEX 9 costs $190 on 5" disk $205 on 8" disk
Version 3.2 for FLEX 2 costs $150 on 5" disk $165 on 8" disk

*FLEX and UniFLEX are trademarks of Technical Systems Consultants



The ROM PACKAGE of utilities (from $250) enables you to install your program in EPROM and includes a license fee for inclusion of the ROM run-time system in an OEM product.

INCLUDE ($25 + media charge) is a pre-processor for Pascal text enabling you to manage your source code libraries easily.

XREF ($25 + media charge) keeps track of the variables in your Pascal (or any other) source file by generating a full cross-reference listing.

PROFILER ($25 + media charge) processes Pascal source code to show the static block structure. Invaluable for analysing or re-structuring other people's programs!

INCLUDE, XREF and PROFILER are supplied in source form and can be provided on a single disk. The media charge is $15 for 8" or $25 for 8" disk.

COPYCAT is a collection of software utilities enabling you to read mini-FLEX, SSB DOS68 or CP/M disks. It is supplied in source form and comes complete with a manual for $50 on 5" or $65 on 8" disk.

TEKPAK enab s you to implement graphics on your system. It offers most of the features of Tektronix PLOT10 software which control Tektronix 40xx graphics terminals, or those emulating them. TEKPAK is supplied in source form, with a manual and demonstration programs for $100 on 5" or $115 on 8" disk.

All Lucidata products are fully backed by our 90 day Warranty and our Update Service.

All Lucidata prices include postage and packing anywhere in the World. They only apply to Cash-with-Order and credit card sales. We accept VISA and MasterCard. Products are normally dispatched within 24 hours by First Class Airmail (UK to USA about 5 days). Please specify your system when ordering or sending for data sheets.

LUCIDATA LTD. P.O. Box 128 CAMBRIDGE CB2 5EZ ENGLAND TELEPHONE (0223) 841906

CLASSIFIED ADVERTISING

For Sale: Inexpensive starter system. Thomas Instruments Super CPU, Thomas Instruments video board, D.S.D. 16K RAM, JPC high-speed cassette interface, full-featured programmable keyboard, SWTP chasis, heavy duty power supply, many interesting extras. $450.
Bob Levine, 32 King St., New York, NY 10014; (212) 691-2897.

***

SWTPC 6800 System: 32K + 4K board, MP-C, 2 MP-S, MP-LA, JPC cassette interface w/CFM3, AC-30, CT-64, HEATH H14 printer. Software: (3 Basics, Pascal, Text Editor), and documentation. First reasonable offer accepted.
Scott Quackenbush, Box 275, St. John, Ks 67576.

***

SWTP 6800/2 System, 28K, MP-S, MP-LA, 2 unpopulated 8K memory boards, AC-30. $325.
Bill Kramer, (612) 774-4971 eve.

***

ET-3400 Microprocessor Trainer $75. ETA-3400 Memory Add-On I/O Accessory $75. Both for $125 !! I will pay postage.
George Kelm, PO Box 160, Yap, Caroline Is., TT (Guam) 96943. US PO Domestic Zone 8.

***

SWTPCo 6800/2 case $15, MP-B $25, MP-LA $20, MP-S $20, MP-T $45, AC-30 $35, SMS DM-64 64K memory board less memory IC's $100, F&D PMB-1 video board $75, July 80 to Aug 82 68' Micro Journals make offer.
R Richardson, 1810 Byron Creek Rd, Winston Or 97496.

***

For Sale: SS-50 6800-6809 equipment/software, send SASE for list.
Roy Hawkins, W. 341 S. 5065 McCoy Pkwy, Dousman WI 53118.

***

TEN MOST-ASKED QUESTIONS

ABOUT DYNACALC™

THE ELECTRONIC SPREAD-SHEET FOR 6809 COMPUTERS

1. **What is an electronic spread-sheet, anyway?**
   Business people use spread-sheets to organize columns and rows of figures. DYNACALC simulates the operation of a spread-sheet without the mess of paper and pencil. Of course, corrections and changes are a snap. Changing any entered value causes the whole spread-sheet to be re-calculated based on the new constants. This means that you can play, 'what if?' to your heart's content.

2. **Is DYNACALC just for accountants, then?**
   Not at all. DYNACALC can be used for just about any type of job. Not only numbers, but alphanumeric messages can be handled. Engineers and other technical users will love DYNACALC's sixteen-digit math and built-in scientific functions. There's even a built-in sort command, so you could use DYNACALC to manage small data bases - up to 256 records.

3. **What will DYNACALC do for ME?**
   That's a good question. Basically the answer is that DYNACALC will let your computer do just about anything you can imagine. Ask your friends who have VisiCalc, or a similar program, just how useful an electronic spread-sheet program can be for all types of household, business, engineering, and scientific applications.

4. **Do I have to learn computer programming?**
   NO! DYNACALC is designed to be used by non-programmers, but even a Ph.D. in Computer Science can understand it. Built-in HELP messages are provided for quick reference to operating instructions.

5. **Do I have to modify my system to use DYNACALC?**
   Nope. DYNACALC uses any standard 6809 configuration, so you don't have to spend money on another CPU board or waste time learning another operating system.

6. **Will DYNACALC read my existing data files?**
   You bet! DYNACALC has a beautifully simple method of reading and writing data files, so you can communicate both ways with other programs on your system, such as the Text Editor, Text Processor, Sort/Merge, RMS data base system, or other programs written in BASIC, C, PASCAL, FORTRAN, and so on.

7. **How fast is DYNACALC?**
   Very. Except for a few seldom-used commands, DYNACALC is memory-resident, so there is little disk I/O to slow things down. The whole data array (worksheet) is in memory, so access to any point is instantaneous. DYNACALC is 100% 6809 machine code for blistering speed.

8. **Is there a version of DYNACALC for MY system?**
   Probably. You need a 6809 computer (32k minimum) with FLEX or UniFLEX operating system. A version for OS-9 is also in the works. You also need a decent CRT terminal, one with at least 80 characters per line, and direct cursor addressing. If your terminal isn't smart enough for DYNACALC, you probably need a new one anyway. The UniFLEX version of DYNACALC also allows you to mix different brands of terminal on the same system. There's also a special version of DYNACALC for Color Computers equipped with FLEX and Data-Comp's F-MATE. A version for Frank Hogg's Color Computer FLEX is also being done.

9. **How much does DYNACALC cost?**
   The FLEX versions are just $200 per copy; UniFLEX version $395. Foreign orders add $10 per copy for postage. We encourage dealers to handle DYNACALC, since it's a product that sells instantly upon demonstration. Call or write on your company letterhead for more information.

10. **Where do I order DYNACALC?**
    See your local DYNACALC dealer, or order directly from CSC at the address below. We accept telephone orders from 10 a.m. to 6 p.m., Monday through Friday. Call us at 314-576-5020. Your VISA or MasterCard is welcome. Please specify diskette size for FLEX versions. Software serial number is required for the UniFLEX version of DYNACALC.

ORDER YOUR DYNACALC™ TODAY

Foreign Dealers:

Australia & Southeast Asia: order from Paris Radio Electronics, 7A Burton St., Darlinghurst, NSW 2010 Sydney. Telephone: 02-357-5111.

United Kingdom: order from Compusense, Ltd., PO Box 169, London N13 4HT. Telephone: 01-882-0681.

Scandinavia: order from Swedish Electronics hk AB, Murargatan 23-25, Uppsala S-754 37 Sweden. Telephone: 18-25-30-00.

Computer Systems Center

13461 Olive Blvd.

Chesterfield, MO 63017

(314) 576-5020

UniFLEX software prices include maintenance for the first year.

DYNACALC, DYNAMITE, and DYNAMITE + are trademarks of Computer Systems Center.

F-MATE is a trademark of Data-Comp.
VisiCalc is a trademark of VisiCorp.
OS-9 is a trademark of Microware and Motorola.
FLEX and UniFLEX are trademarks of TSC.

ALSO FROM CSC

DYNAMITE +

"THE CODE BUSTER"

now available for UniFLEX
OS-9 version soon

DYNAMITE + is a new version of DYNAMITE, our popular 6809/6800 disassembler package for 6809 FLEX. Present users of DYNAMITE can upgrade to DYNAMITE + by sending us the original DYNAMITE diskette and $40 (plus $5 for foreign postage). DYNAMITE + does everything DYNAMITE does, and more! A cross-reference generator has been added, label files are now maintained only in text form (LABEL EQU $xxxx), and boundary file specifications have been tremendously simplified, which makes it easier to disassemble large programs containing lots of big tables.

The UniFLEX version of DYNAMITE + does everything the FLEX version does, and also automatically handles system calls and 'info' areas.

DYNAMITE + is available for $100 per copy on FLEX (specify diskette size), and $300 on UniFLEX. Foreign orders add $5 per copy for postage.

SCREEN EDITOR/WORD PROCESSOR FOR DOS6x, FLEX, OS-9

IF YOU ARE CONSIDERING UPGRADING YOUR EDITING AND PROCESSING CAPABILITIES, HERE ARE A FEW GOOD REASONS TO CONSIDER SCREDITOR...

REASON #1 - SCREEN EDITING

Screen editing (sometimes called cursor-based editing) is the most efficient way in which to enter your thoughts into the computer. Since you see the actual text you are entering on the screen as if you were looking at a printed page, organization becomes a matter of observation rather than guesswork.

REASON #2 - DYNAMIC SCREEN FORMATTING

With dynamic screen formatting, the screen becomes an image of what the completed job will look like. Within the capabilities of the display terminal itself, you are constantly looking at what you would have if your job were printed. Every change you make, every addition or deletion, is automatically incorporated in a fully-justified, ready-to-print form. You no longer need to guess what all those cryptic processing commands you may have used with other software will turn your finished job into!

REASON #3 - NO MACHINE LANGUAGE PATCHING

SCREDITOR III is unique in that you NEVER have to make machine language patches to integrate it into your system. If you can answer a series of questions about your printer and terminal's capabilities, installation is complete.

REASON #4 - NO EXTRA "HIDDEN" PROGRAM COSTS

SCREDITOR III is complete as purchased. You get screen editing, formatting, mailing list merging and Mouth Mate, all in a single program. And it up. Even though the competition can provide many of the features of SCREDITOR III, how much will you pay? And, more important, how many programs will you have to buy?

REASON #5 - OPERATING SYSTEM SUPPORT

To our knowledge, SCREDITOR III supports more processor and operating system combinations than ANY other single editing and word processing package available. OS-9 (level one and two), Flex and SSB DOS, 6800 and 6809 versions are available. Except for those commands which are peculiar to a particular operating system, SCREDITOR III is identical in operation in all versions.

REASON #5 1/2 - OPERATING SYSTEM SUPPORT COSTS

If you purchase another company's product for your FLEX-9 system and decide to run OS-9, how much will it cost you for the extra copy of their software (IF it's available. See reason #5.)? With SCREDITOR III, tell us you are upgrading and you pay 40% of the price you paid for the first copy and you can have the same great capabilities with both operating systems!

REASON #6 - CAPABILITIES

Not only is SCREDITOR III the only COMPLETE editing and word processing program available, it's also the most powerful. See if the other combinations of programs available give you true multi-column editing. Or allow you to add and decimal align rows and columns of numeric data. Or allow you to change margins and tabs at any time during editing just like a typewriter. Or give you commands to re-justify and change the width of paragraphs and lines on-screen, while you watch. Or edit and re-execute commands. Or allow essentially unlimited stored text and command entry replacement with a single keystroke (macros). Or let you save all of those macros to a data file to be re-loaded for use in later edit sessions. Or edit your macros. Or...

REASON #7 - DOCUMENTATION

SCREDITOR III has the most comprehensive documentation available for any 6809 editor/word processor. The Technical Reference Manual is currently about 160 pages long. The Training and User's Guide we are now preparing will be about another 100 pages. We're also working on our new Tutorial Cassette (two hours of instruction). We even provide a HELP command which gives you the syntax of all of SCREDITOR III's commands and at least one paragraph of description on each!

REASON #8 - PRICE

Considering its capabilities, SCREDITOR III is the least-expensive program of its type available today, Period.

REASON #9 - HAPPINESS

We want your business. Think how happy you will make us if you decide to buy SCREDITOR III!

The following information should answer most of your questions about SCREDITOR III. If you have additional questions, please call during our office hours, or, better still, order the manual (when you decide to buy SCREDITOR III, we'll apply your manual purchase against the software price).

PROCESSORS SUPPORTED - 6800, 6809

OPERATING SYSTEMS - FLEX, SSB DOS, OS-9 (level 1 and 2)

DISPLAY REQUIREMENTS - Serial terminal with computer-addressable cursor. 4800 baud minimum data rate, 9600 or better preferred. Clear EOL and clear EOS not required but highly desirable. Insert and Delete line improve display speed. Suggested terminals include Televideo, Beath/Zenith H-19, SWTPCo. Virtually all addressable-cursor terminals can be used. ANSI terminals supported by the time you read this.

Memory-mapped displays require software to provide terminal look-alike capabilities. Sample software included with FLEX and DOS6x orders.

Minimum display size is 16x64. Maximum is 255x255. No software patching is required to adapt software to display size.

Unique configurations may be supported by calls to external user-supplied sub-routines.

KEYBOARD REQUIREMENTS - ASCII character set with control codes required. Function keys generating multiple-character sequences also supported. Control code and function key assignments user-defined and changeable at any time without patching.

KEYBOARD INTERFACE - PIA parallel, ACIA serial, or external keyboard handlers. Interrupt-driven operation (queue-buffered type-ahead) is a standard user-selected option.

PRINTER HARDWARE - Current version supports standard printers. Specialty printer capabilities controllable by transmission of control or escape sequences imbedded in text. Underline, boldface, double width simulated on printers with backspace capability. Proportional printing available soon.

PRINTER INTERFACE - Program uses existing user-supplied software to transfer characters from SCREDITOR III to printer hardware by calls to user sub-routines.

COMMAND SET - Forty-six commands (or more by the time you read this) with multiple options on many commands. Thirty-one single key commands for often-used operations such as cursor movements, inserts, deletes, paragraph formatting, etc...

OPERATION AND DISPLAY - Text entry by 'window-type' screen/cursor editing. Commands by single-key or line entry. Continuous display of current tabs/margins, edit modes, cursor row and column, current line number, horizontal display range.

REQUIRED MEMORY - (program memory) - Program size - approximately 20 kilobytes. Minimum memory for useful editing - 32 kilobytes.

DISK MEMORY - Configuration files (3 minimum) - 1 to 2 sectors each
HELP file - disk-resident, approximately 19 kilobytes

********** AT LAST! **********

AVAILABILITY - Please allow one to two weeks for delivery. Disks are built as orders are received to allow the most current possible version of software to be shipped.

PRICE - SCREDITOR III manual, disk $150.00
manual only (price may be applied to software order) . . . . . $ 40.00

IMPORTANT INFO

We accept Mastercharge and Visa credit card orders, personal and business checks drawn on U.S. banks and in U.S. dollars, and COD's to the continental United States. Domestic orders shipped by United Parcel Service unless otherwise requested. Overseas orders shipped by air parcel. COD's shipped by United Parcel Service only. Open accounts to government agencies, educational institutions, established accounts and Fortune 1000's only. Virginia residents add 4% state sales tax. On orders under $100, add $5 for shipping and handling. On overseas orders add $15 for shipping.

THE BEST NEWS!

In our haste to do business, too often the temptation to compromise our faith weighs heavily. Even though we have forgiveness when we confess our sins from a God who loves us, we would like to take a moment to ask forgiveness from any of our customers whom we have slighted. Orders do get lost or mis-shipped, and mistakes do happen. If we fail to completely please any of you in either our response to your order, inquiries, or for whatever reason, let us know. We will do anything in our power to satisfy your problems. We thank all of our customers for your prayers, and we pray for each of you that you will come to know the love of God in your lives that we have found!

ALFORD & ASSOCIATES
P.O. BOX 6683
RICHMOND, VA., 23230
804-320-6722

CoCo SPEECH SYNTHESIZER

The Color Computer is one of the best things to happen to computing since the SS-50 bus! At the price, there is no better way to get into personal computing, and we are now offering a speech synthesizer for the CoCo! With most of the same features of our VS-1, the SP-1 can give your CoCo the power of unlimited speech. Unlike software approaches to speech, the SP-1 does not eat your memory, and provides the clearest, easiest to understand speech we have heard!

The SP-1 is a completely assembled and tested unit which plugs directly into the MINI-PACK slot on the side of the computer. Unlike the VS-1, however, no external speaker is required, as the SP-1 routes its audio directly through the computer and into your television monitor!

We also have a version of our VOS-EDITOR to let you build your voice messages quickly and easily, and then save them to cassette when you are done. The speech files can later be read by Basic in your application programs! The comprehensive manual we supply gives you all the information you need to make your CoCo talk!

SP-1 pack and manual $179.95
SP-1 manual only (price applied to SP-1 purchase) . . . . . $ 19.95

TRANSLATOR FOR SP1

Our SC-1/COLOR translator uses the same highly accurate algorithm as our SC-1 for the VS-1, but drives the SP-1 on the color computer. The SC-1/COLOR, however, is even easier to use! When the SC-1/COLOR translator is loaded, it becomes a part of Basic! To translate and speak English text, numeric data (even punctuations if you desire), you simply program PRINT #2 followed by your text. The translator takes it, converts it to a phoneme stream, and drives the SP-1 directly!

SC-1/COLOR cassette and manual (requires SP-1 for operation) $69.95
(manual not available separately)

SSB UTILITIES DISK

Two of the shortcomings of SSB DOS are that, one, with six-character file names and default extensions it's hard to know from a name what is in a file, and, two, with no disk information sector the only way to know what is on a disk is to list the files! Since we couldn't conveniently make the file names longer, we did the next best thing and wrote a family of programs to allow you to "name" your disks!

Utilities #1 provides a NAME utility to build a disk information sector on each newly-formatted disk. UPDATE lets you change this sector, LOCK lets you write, delete or even LIST protect the files on the disk. LIST is a better listing utility which reads the disk access code from the information sector and will only list LOCK'ed files if you supply the access code. A number of other programs are also included on the disk.

The manual provides information about the way SSB disks are formatted and how the chained sector method work in addition to complete descriptions about each of the programs provided on the disk!

UTILITIES #1 disk and manual $29.95
(manual not available separately)

FANTASTIC PROOFREADER

We are proud to offer Pete Stark's Spell 'n Fix proofreader. We examined a number of very good pieces of spelling correction software. We compared ease of use, speed, correction capability, dictionary maintenance and expansion , disk space overhead, etc..., and found that Pete Stark of StarKits has done a remarkable job on this program. Look at these features and make your own decision....

MULTIPLE DICTIONARIES - Spell 'n Fix allows you to create multiple dictionaries. In this way, if you want, you can put technical and special words in a separate dictionary. Then, when you proofread your documents, select the dictionaries which have the words you need, thus speeding up the job!

EXECUTE DOS COMMANDS - Need to list your disk to decide which dictionary or text file to proofread? Want to delete an old file? View a file? Most of your DOS commands can be executed directly from within Spell 'n Fix.

MULTIPLE CORRECTION OPTIONS - With Spell 'n Fix, you can add new words to your dictionary at any time. Or, if you want, you can create a new dictionary for words not found in the existing dictionaries. Or simply print out the new words. Or even ignore them! CORRECTION OR MARKING - With Spell 'n Fix, you can mark incorrectly-spelled words in your text for later correction, or you can tell Spell 'n Fix to correct them as they are found. The choice is yours!

COMPRESSED DICTIONARY - Spell 'n Fix uses either of two forms of dictionary, and converts either to the other at your command. The first type is a fully expanded dictionary with one word per line, allowing you to use your editor to make corrections, deletions and additions easily. The second type incorporates a slick compression method to drastically reduce dictionary size and increase access speed.

After looking at our options, we decided that this was the program for us, and are proud to be the exclusive agents for StarKits for the SSB DOS6x and DOS09 versions. We can also supply versions for 6800 and 6809 Flex. The OS-9 version will be available soon directly from StarKits. StarKits also has a version for the Color Computer! At the price, we feel that no other program can do what Spell 'n Fix can....

SPELL 'N FIX disk (with 17000 word dictionary) and manual $69.29
(manual not available separately)

SS50 SPEECH SYNTHESIZER

The VS-1 is a completely-assembled, tested unit for the SS-50 I/O bus which gives your computer the power of unlimited speech. Since the VS-1 is programmed phonetically, there is essentially no limitation on what you can make your computer say! The VS-1 programs just like a parallel printer driver, and requires an average of only one phoneme per letter in English for a very low message memory storage overhead. The data rate is typically a low 70 baud.

The VS-1 is supplied with the easiest-to-use, most flexible speech editing program available. This program makes building your speech files or data a real pleasure. We also supply additional software and sample speech files on the disk to give you a 'leg up' on making your computer talk.

The manual provides sections on speech generation theory, operation of the supplied software, complete technical descriptions, a sample speech dictionary and phonetic code charts, fifty pages in all!

If you are a roll-your-own nut, we provide the bare board or manual or disk or synthesizer chip, or combinations (We suggest that only VERY experienced persons attempt such assembly due to the fact that we do not provide assembly instructions beyond parts list parts layout and schematic).

The VS-1 is available for 6800 and 6809 FLEX, 6800 and 6809 SSB DOS, and OS-9 level I and II.

VS-1 complete with manual and disk $194.95
VS-1 manual and disk only (may be adapted for brand-X) . . . . . $ 29.95
VS-1 manual only (may be applied to VS-1 order. Look before you leap!) . . . . . $ 19.95
VS-1 bare board (build your own...requires experience!) . . . . . $ 19.95
SC-01A synthesizer chip . . . . . $ 60.00
VS-1 bare board, chip, disk and manual . . . . . $ 89.95

TRANSLATOR FOR VS1

If you have our VS-1 (or even brand-X), and want an easier way to make your computer talk, our SC-1 translator is for you. The SC-1 translator takes English text from a disk file or your keyboard, translates it to a phoneme stream, and sends the resulting output to either the speech synthesizer hardware or to a data file for later use.

The SC-1 translator is one of the most accurate such programs we have seen, making typically only one 'mistake' per sentence, for an average accuracy of about 95%! Due to the type of algorithm we are using, even proper names are generally translated correctly! Where the SC-1 is being used to compile 'canned' messages for later use, we provide an additional program which takes a binary phoneme file created by the SC-1 and converts it to a speech file which may be edited by our VOS-EDITOR (supplied with the VS-1). In this way, you can quickly create your messages and then 'fine-tune' them!

The SC-1 translator is available for 6800 and 6809 FLEX, 6800 and 6809 SSB DOS, and OS-9 level I and II. If you are using brand-X, order the SC-1 with the VOS-EDITOR on the same disk!

SC-1 disk and instructions $39.95
SC-1 disk, instructions with VOS-EDITOR $40.95

TREK-68

Our TREK program is the best 'real-time', 'live-action' game available for 68xx computers using memory-mapped displays. This program does not require graphics capabilities from your display. Otherwise, this is a full arcade-type game. No triviality here, either. The suicide option has only been won by a handful of people world-wide, and can run for hours!

TREK is available for 6800 and 6809 FLEX and 6800 and 6809 SSB DOS, and requires both a memory-mapped display and a source of 60 or 100 Hz interrupts (such as the MP-T timer board).

TREK disk and manual $24.95

COLOR-TREK

Like our other version of TREK, COLOR-TREK provides you seat-of-the-pants action. This program uses no routines from Basic, and so can run on non-extended Basic machines. 16k memory is required. Only one player to our knowledge (other than the author) has won the suicide option of this game, and it took him fifty-seven minutes! No triviality here!

COLOR-TREK cassette and manual $17.95

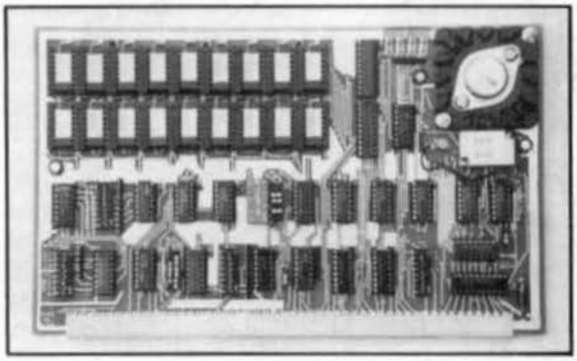
WINCHESTER FOR MOTOROLA EXORCISOR/MDOS

☐ 10 MB Winchester hard disk runs MDOS on Motorola Exorcisor System ☐ No modification to MDOS required ☐ MDOS based software stays alive ☐ All user software operates without modification ☐ Optional SA-801R flexible diskette drive system ☐ Optional 10 MB removable cartridge.

For information call (714) 566-3911

Computer System Associates

7562 Trade Street, San Diego, CA 92121



QMM1-B 256K MEMORY FOR SS50-C 6809 SYSTEMS

Compatible with systems by SSB, GIMIX, and SWTPC including those with DMA disk controllers.

Full 2Mhz operation with transparent on board refresh, runs continuously at 2 Mhz without cycle stretching or stealing.

Very versatile addressing and disable features.

Parity option halts processor and sounds audible alarm upon detecting a read error.

All boards assembled, tested, burned in and warranted for 1 year.

Also available with 64K, 128K, or 192K.

256K for $1135.00 — 256K w/ parity $1270.00

Delivery: Stock—2 weeks. Terms: Prepaid or COD.

D.P. Johnson (503) 244-8152
7655 S.W. Cedarcrest St., Portland, OR 97223

OS9 Application Software

Specialty Electronics, Inc.

ACCOUNTS PAYABLE

Take the "headache" out of tracking your accounts payable with Specialty Electronics interactive Accounting system. The accounts payable package supports these outstanding features:

1. Entry of debits, credits, regular invoices, full and partial payments.
2. Hand checks entered directly or computer printed checks with audit and check ledger.
3. Provides aging of accounts and proper general ledger distribution of all entries.
4. Job costing, customer order number tracking and buyer identification are provided.
5. Vendors may be added as needed.
6. Complete audit trails are provided.
7. Reports can be generated for specific due dates, customer groups, open or closed items and in either voucher detail or vendor summary formats.

Accounts Payable I-code

$299

GENERAL LEDGER with CASH JOURNAL

The general ledger is the center of the Specialty Electronics Accounting System. With the package you can:

1. Produce balance sheets and income statements in various formats.
2. Define account names, spacing, positioning, headings and subaccounts.
3. Format special reports and print percentages.
4. Post by hand cash journal or by using the interactive accounts receivable, payable and payroll.
5. Provides a clear audit trail for all entries.
6. Input data in an easy to follow format.
7. Use for multi company accounting without modification.

General ledger I-code

$399

ACCOUNTS RECEIVABLE

Your Accounts Receivable can be followed with a minimum of time investment using these features:

1. Regular invoicing, debit and credit memos, full and partial payments.
2. Progressive billing and payments.
3. Aging at periods specified by the user.
4. New customers entered as needed.
5. Statements are generated listing individual invoices and overdue amounts, followed by aging category.
6. Total interaction with the general ledger with tax, shipping and travel expenses computed separately and posted to various accounts.

Accounts Receivable I-code

$299

INVENTORY

The Specialty Electronics interactive Accounting System inventory Control Package provides the tools for complete control of a large and active inventory providing:

1. Reports for quantities on hand, quantities on order, activity and many other categories.
2. Complete item description, category groups, supplier information, order dates, reorder quantities, etc.
3. Simple input and recognition procedures.

Inventory Control I-code

$299

PAYROLL

The Specialty Electronics Interactive Accounting System provides payroll support which goes beyond writing paychecks. Its features include:

1. Weekly, biweekly, semimonthly and monthly pay periods.
2. Hourly, salary, vacation, holiday, commission, overtime and compensatory pay categories and tips.
3. Deducts federal and state payroll taxes, insurance, additional taxes, special deductions.
4. Daily time keeping allowed.
5. Prints checks, stubs, check ledger and journal history.
6. Prints W-2 forms, federal and state tax report information.
7. Keeps full employee history.
8. Tax tables allow user modification.

Payroll I-code

$425

Complete Documentation $19.95

* OS9 and Basic 09 are trademarks of Microware, Inc. and Motorola Corp.

P.O. Box 541
2110 W. Willow

Specialty Electronics

(405) 233-1632
Enid, OK 73701

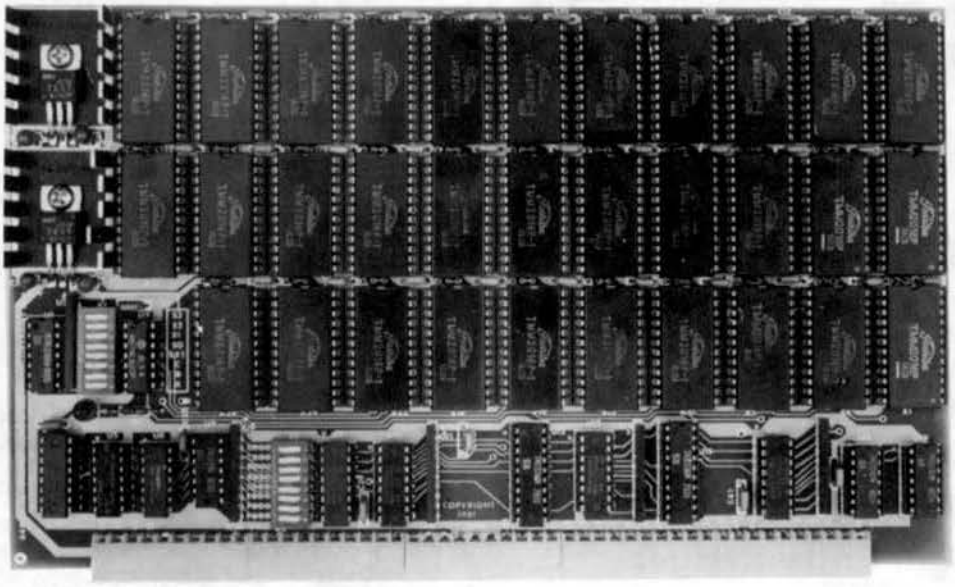
64K SS-50 STATIC RAM

NEW!

$259.00 (48K KIT)

NEW!

LOW POWER!



RAM OR EPROM!

BLANK PC BOARD WITH DOCUMENTATION $52

SUPPORT ICs + CAPS -	$18.00
FULL SOCKET SET -	$15.00

ASSEMBLED AND TESTED **ADD $40**

FEATURES:

- ★ Uses new 2K x 8 (TMM 2016 or HM 6116) RAMs.
- ★ Fully supports Extended Addressing.
- ★ 64K draws only approximately 500 MA.
- ★ 200 NS RAMs are standard. (TOSHIBA makes TMM 2016s as fast as 100 NS. FOR YOUR HIGH SPEED APPLICATIONS.)
- ★ Board is configured as 3-16K blocks and 8-2K blocks (within any 64K block) for maximum flexibility.
- ★ 2716 EPROMs may be installed anywhere on Board.
- ★ Top 16K may be disabled in 2K blocks to avoid any I/O conflicts.
- ★ One Board supports both RAM and EPROM.
- ★ RAM supports 2MHZ operation at no extra charge!
- ★ Board may be partially populated in 16K increments.

56K Kit	$309
64K Kit	$359

16K STATIC RAMS?

The new 2K x 8, 24 PIN, static RAMs are the next generation of high density, high speed, low power, RAMs. Pioneered by such companies as HITACHI and TOSHIBA, and soon to be second sourced by most major U.S. manufacturers, these ultra low power parts, feature 2716 compatible pin out. Thus fully interchangeable ROM/RAM boards are at last a reality, and you get BLINDING speed and LOW power thrown in for virtually nothing.

Digital Research Computers
(OF TEXAS)
P.O. BOX 401565 • GARLAND, TEXAS 75040 • (214) 271-3538

TERMS: Add $2.00 postage. We pay balance. Order under $15 add 75¢ handling. No C.O.D. We accept Visa and MasterCharge. Tex. Res. add 5% Tax. Foreign orders (except Canada) add 20% P & H. Orders over $50, add 85¢ for insurance.

AAA Chicago Computer Center

ELEKTRA CABINET Made of heavyweight 0.090" thick aluminum. Interior is 18.1/2" wide by 21.7/8" deep by 6-3/4" high. Heavy duty A.C. line cord. A.C. fuse holder. EMI filter. Fan with filter. Back panel has 10 cutouts for 'D' type data connectors. Front panel has key on/off power switch, 2 illuminated push button switches (Reset and NMI/Abort), and two cutouts for 5-1/4" disk drives.

POWER SUPPLY Highest quality linear power supply CONSERVATIVELY rated at 15a @ 8v, 3a @ 16v, 3a @ -16v. 3 primary inputs for light, rated, and heavy loading.

DISK REGULATOR BOARD WITH CABLES Standard version for 2 floppy drives. Heavy duty version for 1 Winchester drive and 1 floppy drive.

ELEKTRA MOTHERBOARD Heavyweight 0.125" thick, 18" long by 9" wide. 11 memory (50 pin) slots. 4 or 8 slots may be cut off for shortening to 14" or 10" lengths respectively. 8 I/O (30 pin) slots. Complete address decoding of I/O slots. Choice of 4, 8, or 16 addresses per I/O slot. Base address for I/O slots can be placed at 32, 64, or 128 byte increments respectively. 1" spacing between all memory and I/O slots. Extended addressing capability for both memory and I/O ports for meeting SS-50C bus specifications. On board baud rate generator with low and high ranges providing jumper selectable rates of 75 through 38,400 for each of the five baud rate lines. Slow device circuitry permitting 1MHz 30 pin disk controllers to run with 2 MHz 50 pin CPU boards.

ELECTRA CPU 8/9 Use either the 6802 or 6808 (to run 6800 software) or 6809. Has provision for up to 3 2716 Eproms, 1K scratchpad, MC6840 triple timer, and an optional baud rate generator providing baud rates from 110 through 38,400 baud in two user selectable ranges. The board supports DMA by either HALT or BUSREQ when a 6809 CPU is used. The board does not support a DAT and therefore does not support extended addressing. The board will run any of the MIKBUG™ compatible monitors in the 6802/6808 mode and SBUG-E, HUMBUG, and GMXBUG-09 in the 6809 mode. The ELEKTRA CPU 8/9 will run any of the popular disk controller boards with the appropriate software. OS-9™ Level 1 is available.

ELEKTRA DPS DUAL PORT SERIAL CARD Fits the standard 30 pin SS-50 bus I/O slot. Can be configured for 4 addresses per port with the B port 2 addresses higher than the A port or for 16 addresses per port with the B port 4 addresses higher than the A port. Each port is terminated at two 16 pin dip sockets, one socket configured for modem and the other for terminal or printer. RTS, CTS, DTR, DCD, and DTR are appropriately implemented. Each port has independent selection of baud rate. Each port allows the interrupt request to be independently jumpered to the IRQ or FIRQ/NMI bus line.

ELEKTRA DPP DUAL PORT PARALLEL CARD Fits the standard 30 pin SS-50 bus I/O slot. Can be used in either the 4 or 16 addresses per I/O slot configuration occupying the first four addresses of the I/O slot. The direction of the TTL buffers can be controlled by either on board jumper connectors or by a signal from the peripherals. The interrupt request lines for each port may be individually jumpered to either the IRQ or FIRQ/NMI bus line.

ELEKTRA CHASSIS Includes cabinet, 110v power supply, power supply cables, standard disk regulator board with power cables, assembled motherboard with gold square pin connectors $825.00*

ELEKTRA CABINET	260.00
ELEKTRA 110v POWER SUPPLY	175.00*
ELEKTRA CABINET, 110v POWER SUPPLY, AND POWER SUPPLY CABLES	410.00*
ELEKTRA CABINET, 110v S, S CABLES, STD. DISK REG LATOR AND CABLES	480.00*
*Add $30.00 for 220v.	
STANDARD DISK REGULATOR BOARD AND CABLES	50.00
HEAVY DUTY DISK REG LATOR BOARD AND CABLES	75.00
FILLER PLATE FOR 5-1/4" DRIVE OPENING	10.00
FAN FILTER	10.00
GOLD 10 PIN CONNECTORS (Specify male w/square pins or female)	1.50
TIN 10 PIN CONNECTORS (Specify male w/square pins or female)	.50

	BARE BOARD	KIT	ASSEMBLED
ELEKTRA MOTHERBOARD (Gold connectors)	80.00	320.00	380.00
ELEKTRA MOTHERBOARD (Tin connectors)		240.00	300.00
ELEKTRA CPU 8/9	50.00	225.00	275.00
ELEKTRA CPU 8/9 with baud rate option		250.00	300.00
ELEKTRA DPP DUAL PORT PARALLEL BOARD**	20.00	60.00	80.00
ELEKTRA DPS DUAL PORT SERIAL BOARD**	20.00	60.00	80.00
**CABLE FOR DPP OR DPS (2 needed, specify board)			20.00

WARNING AAA Chicago Computer Center does not provide repair or diagnostic service for customer assembled kits. AAA Chicago Computer Center does warranty and maintain service for our assembled boards. The customer should carefully take into consideration the small differential separating our kit and assembled prices when making his choice of purchase.

We have introduced our line of computer equipment with the purpose of offering the highest quality of components possible at affordable prices. These products are intended for OEM applications where it is the responsibility of the purchaser to integrate these components with suitable memory, disk controllers, drives, and software along with I/O terminals to form working computer systems.

AAA CHICAGO COMPUTER CENTER

120 CHESTNUT LANE • WHEELING, IL 60090

(312) 459-0450

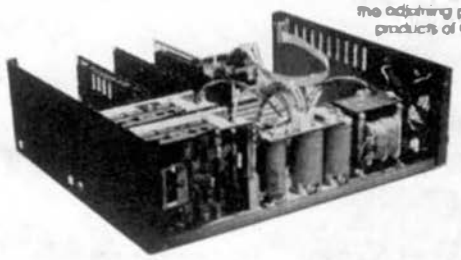
Technical consultation available 4 PM to 6 PM most weekdays. Closed evenings and weekends.

TERMS Minimum order $20.00. Shipping and handling estimates within the Continental U.S., add 3% (MINIMUM $2.50). Illinois residents add 6% sales tax. We will refund your overestimated shipping and handling charges. Foreign shipping and handling, add 10% (MINIMUM $10.00). Foreign orders must be prepaid in U.S. dollars. Heavy foreign items will be shipped air freight collect. Please phone between 4 PM and 6 PM weekdays if questions arise regarding shipping fees. Master Charge, Visa, and American Express honored.

Our apology. We are not staffed to answer technical inquiries through the mail. Please phone for technical help during the hours indicated above. The too frequent changing of our inventory and prices makes it uneconomical to publish a catalog. Our ads are intended to serve that purpose. Prices and inventory are subject to change without advance notice.

ELEKTRA
COMPUTER PRODUCTS

The CPU, 56K memory board, and DMA controller board in the adjoining picture are products of GIMIX Inc.



ELEKTRA CHASIS (above) with 2 GIMIX 32K memory boards (64K total) $1300.00

Special: Gimix 6800 chassis (cabinet, power supply, motherboard) $600.0; 2 Remex DS00 8" drives in cabinet with power supply $1000.00; or take both for $1550.00

SMOOTH™ Software

SUPER MODEM PROGRAM

Transmit manually to distant computer

Transmit disk files (text) of any length to distant computer

Receive and save disk files (text) of any length on local disk system. If sending computer does not support an X-on/X-off protocol, then the received files are limited in size by the computer memory.

Tested to transmit and receive text at speeds up to 9600 baud. (CRT terminal must be capable of operating at a baud rate higher than the one the modem is operated at.)

Half duplex option in case distant computer doesn't echo.

Echo option so user can simulate a time sharing system. (Super Modem Program doesn't support auto-answer but the source is provided for those individuals who wish to adapt our program to their special needs.)

Replaces CR with CR/LF (user option) for those using time sharing systems that don't transmit LF's.

Slow disk file transmit (user option) based on character verification for use on time sharing systems to which disk files cannot be sent at speed suggested by the baud rate.

Please specify 6800 or 6809, SSB or FLEX™, 5" or 8".

Manual and disk with both source and object code $75.00

STANDARD MODEM PROGRAM

Same as Super Modem Program above but without ECHO option, CR/LF for CR option, slow disk file transmit option, nor X-on/X-off option. Reception of disk files is limited to those small enough to completely fit within the receiving buffer.

Please specify 6800 or 6809, SSB or FLEX™, 5" or 8".

Manual with instructions, source listing, and flow chart; disk with both source and object code $45.00

Manual with instructions, source listing, and flow chart 25.00

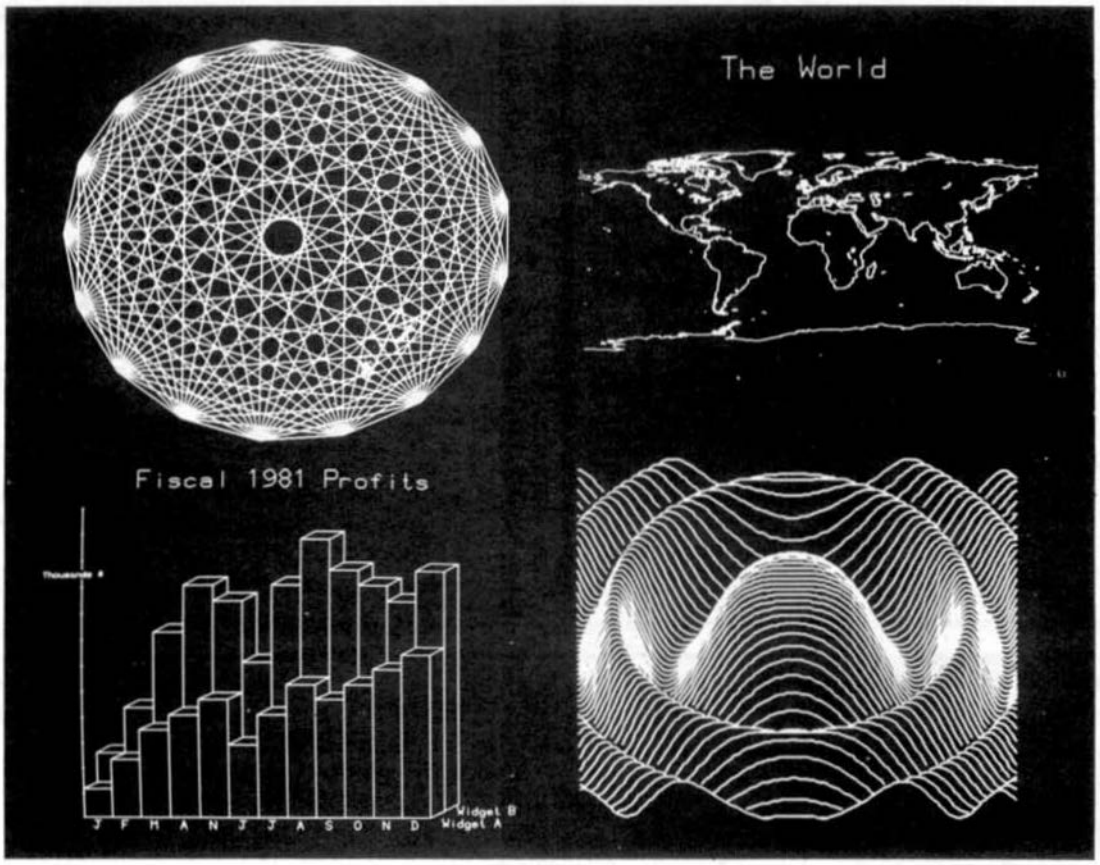
Modems (U.S. Robotics)	Manual Answer	Auto Answer
1200 Baud (120 cps), direct connect	449.00	549.00
300 Baud (30 cps), direct connect	179.00	219.00
300 Baud (30 cps), acoustic	149.00	

ANNOUNCING ElectroScreen™

only $595

the Superior Alternative

to the Traditional Alphanumeric Terminals



The ElectroScreen™ Intelligent Graphics Board Features:

Graphics

- 512 x 480 resolution bit-mapped display
- Interleaved memory access — fast, snow-free updates

Intelligence

- 6809 on-board mpu
- 6K on-board firmware
- STD syntax high level graphics command set
- Removes host graphics software burden
- Flexible text and graphics integration
- Multiple character sizes
- User programs can be run on-board

Terminal

- Terminal emulation on power-up
- 83 characters by 48 lines display
- Easy switching among user-defined character sets
- Fast hardware scrolling

Additional Features

- SS-50C and SS-64 compatible board
- Board communicates with host through parallel latches
- Composite and TTL level video output
- 8 channel 8 bit A/D converter
- Board occupies 4 address bytes

See your dealer today!

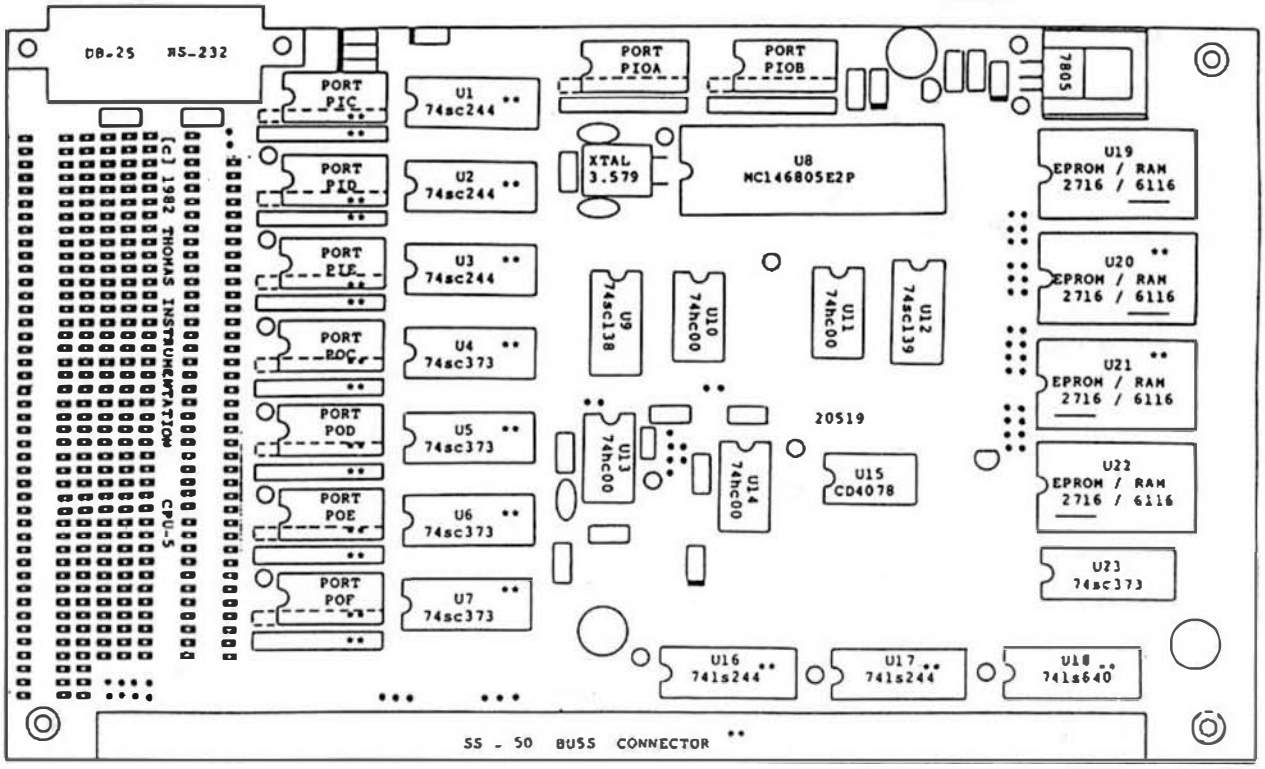
The ElectroScreen manual is available for $10. credited toward purchase of the board.

The ElectroScreen has a 90 day warranty from purchase date.

Dealers, please contact us for our special introductory package.

Privac Inc (703) 671-3900
3711 S. George Mason Dr., Falls Church, Va. 22041

*** NEW 6805 ALL CMOS CARD ***



CPU-5-B	Blank card with documentation	$49.00
CPU-5-P	Partial card (parts marked ** not incl.)	$229.00] Includes EPROM Monitor
CPU-5-F	Full card with all parts shown (fully assembled)	$349.00] and 115 vac Power Pak.
CROSS-ASSEMBLERS	for FLEX ([c] TSC) 6809 system	$150.00

SOFTWARE

VDISK Treat extended memory as a super fast disk drive

6809 source & object	$149.00
6809 object	$ 99.00

OUTSIDE MODEM PROGRAM Incl. source

UniFLEX version	$100.00
FLEX version 6800 & 6809	$ 50.00
HAYES SMART MODEM	$249.00

CROSS ASSEMBLERS for 6800,6801,6805 runs on 6809 FLEX $150.00

TV-EDIT Screen oriented Editor for FLEX 6809 $ 95.00
Please specify 5" or 8" Disk when placing order.

CHIPS

Please note when ordering IC's that a $100.00 Minimum is in effect Orders less than the minimum will be charged an additional $10.00 for handling

6116P-3	2MHZ	$13.50
2016P-2	2MHZ	$10.50
2716	1MHZ	$ 6.50
2716-1	2MHZ	$16.50
146805E2P		$25.00
68802		$10.00
68809		$35.00
68821		$ 6.50
68840		$12.00
68850		$ 6.75
74LS640		$ 3.25

HARDWARE

NEW MEMORY PRICES TO INTRODUCE VDISK

S-R/R without memory chips	$120.00
with 8K NMOS@2MHZ	$169.00
with 16K NMOS@2MHZ	$199.00
with 32K NMOS@2MHZ	$299.00
with 48K NMOS@2MHZ	$399.00
Bare Card	$ 49.00

Extender Cards assemb. with logic ald

SS-50/50C	$ 35.00
SS-30	$ 25.00
SP-1 Prototype Board A/T	$195.00
SP-1 Bare Card	$ 49.00
SS-50 Wire Wrap Board B/C	$ 39.00
SS-30 Wire Wrap Board B/C	$ 20.00
6802 Super CPU A/T	$235.00
6802 Super CPU B/C	$ 59.00
Video RAM, B/C $49.00, A/T	$195.00
Parallel I/O, B/C $49.00, A/T	$139.00
SS-50 Backplanes 4,6,8,12,616 position @ $5.00 per slot w/o connectors	
SS-30 8 pos. BP w/o connect.	$ 39.00
Transition Card $49.00, A/T	$ 95.00
Molex Gold Male $1.60, Female	$ 1.60
Molex Tin Male $ .40, Female	$ .50

THOMAS INSTRUMENTATION

168 EIGHTH STREET — AVALON, N.J. 08202 (609) 967-4280
NJ RES. INCLUDE 5% SALES TAX
Allow 3 Weeks for Checks to Clear
CONT. USA INCLUDE $3.00 SHIPPING, CANADA $6.00, FOREIGN $12.00
MASTERCARD, VISA, and C.O.D. ACCEPTED

Send $20.00 to receive full documentation, schematics, & source listings for all boards currently in production.

'68' MICRO JOURNAL ADVERTISERS INDEX

This index is provided as a reader service. The publisher does not assume any liability for omissions or errors.

THE CHIEFTAIN™ 5¼-INCH WINCHESTER HARD DISK COMPUTER

SO ADVANCED IN SO MANY WAYS . . . AND SO COST-EFFECTIVE . . . IT OBSOLETES MOST OTHER SYSTEMS AVAILABLE TODAY AT ANY PRICE.

● HARD DISK SYSTEM CAPACITY

The Chieftain series includes 5¼- and 8-inch Winchesters that range from 4- to 60-megabyte capacity, and higher as technology advances. All hard disk Chieftains include 64-k memory with two serial ports and DOS69D disk operating system.

● LIGHTNING ACCESS TIME

Average access time for 5¼-inch Winchesters is 70-msec. comparable to far more costly hard disk systems. That means data transfer *ten-times faster* than floppy disk systems.

● 2-MHZ OPERATION

All Chieftains operate at 2-MHz, regardless of disk storage type or operating system used. Compare this to other hard disk systems, no matter ***how*** much they cost!

● DMA DATA TRANSFER

DMA data transfer to-and-from tape and disk is provided for optimum speed. A special design technique eliminates the necessity of halting the processor to wait for data which normally transfers at a slower speed, determined by the rotational velocity of the disk.

● RUNS UNDER DOS OR OS-9

No matter which Chieftain you select . . . 5¼- or 8-inch floppy, or 5¼- or 8-inch Winchester with tape or floppy back-up . . . they ***all*** run under DOS or OS-9 with ***no need*** to modify hardware or software.

● UNBOUNDED FLEXIBILITY

You'll probably never use it, but any Chieftain hard disk system can drive up to 20 other Winchesters, and four tape drives, with a single DMA interface board!

● SMOKE SIGNAL'S HERITAGE OF EXCELLENCE

This new-generation computer is accompanied by the same ***Endurance-Certified*** quality Dealers and end-users all over the world have come to expect from Smoke Signal. And support, software selection and extremely competitive pricing are very much a part of that enviable reputation.

20-Megabyte Tape Streamer Back-Up Option

Available with all Chieftain hard disk configurations. This cartridge tape capability provides full 20-megabyte disk back-up in less than five minutes with just one command, or copy command for individual file transfers. Transfers data tape-to-disk or disk-to-tape. Floppy back-up is also available in a variety of configurations.

The Chieftain Computer Systems:

Here are the Chieftain 6809-based hard disk computers that are destined to change the data processing industry . . .

☐ **CHIEFTAIN 95W4**
4-megabyte, 5¼-inch Winchester with a 360-k floppy disk drive (pictured).

☐ **CHIEFTAIN 95XW4**
4-megabyte, 5¼-inch Winchester with a 750-k octo-density floppy disk drive.

☐ **CHIEFTAIN 98W15**
15-megabyte, 5¼-inch Winchester with a 1-megabyte 8-inch floppy disk drive.

☐ **CHIEFTAIN 9W15T20**
15-megabyte, 5¼-inch Winchester with a 20-megabyte tape streamer.

Write or call today for details (including the remarkably low prices) on the total Chieftain Series . . . and on dealership opportunities.

SMOKE SIGNAL BROADCASTING®
31336 VIA COLINAS
WESTLAKE VILLAGE, CA 91362
TEL (213) 889-9340

Name _____

Company _____

Address _____

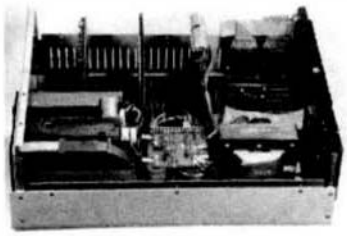
City _____ State _____ Zip _____

Telephone ( ) _____

FLEX - OS-9 LEVEL ONE - UNIFLEX - OS-9 LEVEL TWO

ONLY GIMIX Systems can be configured to run any of these.

GIMIX systems utilize the most powerful 6809 operating systems: FLEX, UniFLEX, OS-9 LEVEL ONE and TWO -- the systems the PROs use. This means a wide selection of software to choose from as well the ability to develop sophisticated, multi-user/multi-tasking programs on your **GIMIX** System.



The GIMIX CLASSY CHASSIS™ consists of a heavy-weight aluminum mainframe cabinet which provides more than ample protection for the electronics and 1 or 2 optional 5¼" drives.

Backpanel connectors can be added for convenient connection of terminals, printers, drives and other peripherals.

A 3 position locking keyswitch enables users to disable the front panel reset button to prevent accidental or unauthorized tampering with the system.

The **GIMIX** system mother board provides fifteen 50 pin slots and eight 30 pin I/O slots -- the most room for expansion of any SS50 system available. The on board baud rate generator features 11 standard baud rates, 75 to 38.4K, for maximum versatility and compatibility with other systems. Extended address decoding allows the I/O block to be addressed anywhere in the 1 megabyte address space. All components feature Gold plated connectors for a lifetime of solid connections. All boards are fully buffered for maximum system expansion.

Each **GIMIX** Mainframe System is equipped with an industrial quality power supply featuring a ferro-resonant constant voltage transformer to insure against problems caused by adverse power input conditions such as A.C. line voltage fluctuations etc. The supply provides 8 volts at 30 amps and plus or minus 16 volts at 5 amps, more than enough capacity to power a fully loaded system and two internal drives.

The 2MHz GIMIX 6809 PLUS CPU board includes a time of day clock with battery back-up and 6840 programmable timer to provide the programmer with convenient, accurate time reference. Later addition of 9511 or 9512 arithmetic processors is provided for on the board. The unique **GIMIX** design enables software selection of either OS-9 or FLEX, both included in many complete **GIMIX** systems.

GIMIX STATIC RAM boards require no complicated refresh timing cycles or clocks for data retention. GIMIX memory boards are guaranteed for 2 MHz operation with no wait state or clock stretching required.

Our low power NMOS RAM requires less than 3/4 amp at 8V for a fully populated 64K board. For critical situations, our non-volatile 64K byte CMOS static RAM boards with built in battery back-up retain data even with system power removed. A fully charged battery will power this board for a minimum of 21 days. A write protect switch permits CMOS boards to be used for PROM/ROM emulation and software debugging.

The GIMIX DMA controller leaves the processor free to perform other tasks during disk transfers - an important feature for multi-user/multi-tasking systems where processor time allocation is critical. The DMA board will accomodate up to 4 drives 5¼" or 8" in any combination running single or double density single or double headed. Programmed I/O Disk Controllers are also available.

GIMIX systems are designed with ultimate **RELIABILITY** in mind. You can choose from the below featured systems or select from our wide variety of components to build a custom package to suit your needs.

GIMIX 2MHz 6809 System including: CLASSY CHASSIS, 6809 PLUS CPU BOARD, 56KB STATIC RAM, 2 SERIAL PORTS W/CABLES, GMXBUG MONITOR, FLEX, and OS-9 LEVEL 1 ..... **$3248.49**
FOR TWO 5¼" 40 TRACK DSDD DRIVES ADD ..... **$ 900.00**
GIMIX 128KB WINCHESTER SYSTEM including: CLASSY CHASSIS, 6809 PLUS CPU BOARD, 128KB STATIC RAM, 4 SERIAL PORTS W/CABLES, 5¼" 80 TRACK DSDD FLOPPY DISK DRIVE, 19MB 5¼" WINCHESTER HARD DISK, OS9 LEVEL 2, EDITOR AND ASSEMBLER **$8998.09**

50HZ Versions Available, 8" Drives Available — Contact GIMIX for Prices and Information.

The Sun Never Sets On A GIMIX!

GIMIX users are found on every continent, including Antarctica. A representative group of GIMIX users includes: **Government Research and Scientific Organizations** in Australia, Canada, U.K. and in the U.S.; NASA, Oak Ridge, White Plains, Fermilab, Argonne, Scripps, Sloan Kettering, Los Alamos National Labs, AURA, **Universities:** Carleton, Waterloo, Royal Military College, in Canada; Trier in Germany; and in the U.S.; Stanford, SUNY, Harvard, UCSD, Mississippi, Georgia Tech. **Industrial users** in Hong Kong, Malaysia, South Africa, Germany, Sweden, and in the U.S.; GTE, Becton Dickinson, American Hoechst, Monsanto, Allied, Honeywell, Perkin Elmer, Johnson Controls, Associated Press, Aydin, Newkirk Electric, Revere Sugar, HI-G/AMS Controls, Chevron. **Computer mainframe and peripheral manufacturers,** IBM, OKI, Computer Peripherals Inc., Qume, Floating Point Systems. **Software houses;** Microware, T.S.C., Lucidata, Norpak, Talbot, Stylo Systems, AAA, HHH, Frank Hogg Labs, Epstein Associates, Softwest, Dynasoft, Research Resources U.K., Microworks, Meta Lab, Computerized Business Systems.

1337 WEST 37th PLACE
CHICAGO, ILLINOIS 60609
(312) 927-5510
TWX 910-221-4055

GIMIX INC.

The Company that delivers

Quality Electronic products since 1975.

FLEX & RS COLOR COMPUTER

If you are tired of playing games on your TRS-80C™ Color Computer, or find that you are handicapped by the limitations of the RS BASIC in trying to write a Program that will allow you to actually USE the Color Computer as a COMPUTER; and if you have been studying the Advertisements in this Magazine and wishing that you could run THESE Programs on your Computer, YOU ARE READY TO MOVE UP TO THE FLEX9™ Operating System. If you want to have REAL PROGRAMMING POWER, using an Extremely Powerful Business BASIC, PASCALS, C Compilers, a full-blown Macro Assembler with a Library capability so you are not continuously "reinventing the wheel", YOU ARE READY TO MOVE UP TO THE FLEX9™ Operating System. If you would like to see if YOU REALLY COULD USE A COMPUTER IN YOUR BUSINESS, or begin to make your Computer start PAYING ITS OWN WAY by doing some Computer Work for the millions of small businesses around you, such as Wordprocessing, Payroll, Accounting, Inventory, etc., then YOU ARE READY TO MOVE UP TO THE FLEX9™ Operating System. How?? DATA-COMP has the way!

DATA-COMP's FLEX9™ Conversion for the TRS-80C™ Color Computer was designed for the SERIOUS COMPUTER USER: with features like greatly increased Display Screens, WITH Lower Case Letters, so you can put a FULL Menu on ONE Screen, or see SEVERAL Paragraphs at the same time; with features like providing a FULL Keyboard so you have FULL Control of your Computer AND it's Programs NATURALLY, without needing a chart to see what Key Combination will give you what function; with USER ORIENTED functions to make using the Operating System natural, like having the Computer AUTOMATICALLY determine what type of Disk is being used in what type of Disk Drive and working accordingly, rather that you have to specify each and every thing for it, or like having the Computer work with the Printer you have been using all along without you having to tell the new Operating System what is there; etc., etc., etc.

DATA-COMP has everything you need to make your TRS-80C™ Color Computer WORK for YOU, from Parts and Pieces to Full, Ready To Use SYSTEMS. DATA-COMP designs, sells, services, and SUPPORTS Computer SYSTEMS, not just Software. CALL DATA-COMP TODAY to make your Computer WORK FOR YOU!

-- SYSTEM REQUIREMENTS --

FLEX9 Special General Version w/Editor & Assembler (which normally sell for $50.00 ea.)	$150.00
F-MATE(RS) FLEX9 Conversion Rout. for the RS Disk Controller when purchased with Special General FLEX9 Sys.	$49.95
when purchased without the General FLEX9 Sys.	$59.95
NEW -- Full Source Code for the Conv. Routines	$159.95
Set of Eight 64K RAM Chips w/ Mod. Instructions	$99.95
Color Computer with 64K RAM and EXT. BASIC	$549.95
Color Computer with 16K RAM	$375.95
Color Computer with 16K RAM and EXT. BASIC	$465.95

-- SPECIAL SYSTEM PACKAGES --

64K Radio Shack COLOR COMPUTER, Radio Shack CO OR DISK CONTROLLER, a Disk Drive System, Special General Version of FLEX9™, F-MATE(RS)™, and a Box of 10 Double Density Diskettes; a COMPLETE, ready to run SYSTEM on your Color TV Set.
$1249.95

-- FOR USERS THAT ALREADY HAVE FLEX9 & Disk Drives --

Radio Shack **DISK CONTROLLER with F-MATE(RS)™** and a Special Two Drive DISK CABLE — $254.95

-- DISK DRIVE PACKAGES, with RS Controller --

These Packages **include** the Radio Shack Disk Controller, Disk Drives with Power Supply and Cabinet, and Disk Drive Cable:

PAK #1 ==> 1 Single Sided, Double Density Sys.	$499.95
PAK #2 ==> 2 Single Sided, Double Density Sys.	$729.95
PAK #3 ==> 1 Double Sided, Double Density Sys.	$579.95
PAK #4 ==> 2 Double Sided, Double Density Sys.	$889.95

-- PARTS AND PIECES --

Radio Shack Disk Controller	$169.95
1 ea. Single Sided, Double Density Disk Drive	$249.95
1 ea. Double Sided, Double Density Disk Drive	$349.95
Single Drive Cabinet with Power Supply	$79.95
Double Drive Cabinet with Power Supply	$99.95
Single Drive Disk Cable for RS Controller	$24.95
Double Drive Disk Cable for RS Controller	$34.95
Micro Tech. Prods., Inc. LOWER CASE ROM Adapter	$74.95
Radio Shack BASIC Version 1.1 ROM	$34.95

SOFTWARE

CHESS 6809

Requires FLEX™ and one of the following CRT terminals

Now Runs On Any Type Terminal

Features:

- **Two display boards.**
- **Four levels of play.**
- **Point scoring system.**
- **Play white or black.**
- **Change or set-up boards/piece positions.**
- **Forfeit move.**
- **Swap sides.**
- **Make move and swap sides.**
- **Change skill level.**
- **Stop and restart game.**
- **Solve 'Mate in 1-2-3-4' moves.**

$79.95 Specify 5" or 8" disk

This is one of the strongest CHESS programs running on any microcomputer, estimated USCF Rating 1600 +.

Note: Personal checks allow 3-4 week delivery.

DIET-TRAC Forecaster

A Diet Planning and Analysis Program

DIET-TRAC Forecaster is a program that plans a diet in terms of either calories and percentage of carbohydrates, protein and fats (C/P/F %) or grams of Carbohydrate, Protein and Fat food exchanges of each of the six basic food groups (vegetable, bread, meat, skim milk, fruit and fat) for a specific individual.

Sex, Age, Height, Present Weight, Frame Size, Activity Level and Basal Metabolism Rate for normal individuals are taken into account. Ideal weight and sustaining calories for any weight of the above individual are calculated. When a weight goal is given (either gain or loss), and a calorie plan is agreed upon between the computer and the individual, the number of days to reach the weight goal is projected. The starting and ending rate of weight loss is calculated, and a daily calendar with each day's predicted weight for a 30-day period is printed.

FLEX VERSION	**$59.95**
UniFLEX VERSION	**$89.95**

PRINTERS

The Epson MX-80
$495.00
The Epson MX-100
$725.00

MX-70 $355.00 MX-80 FT $575.00

MEMORY

SWTPC-Motorola, MP32
32K Dynamic Memory Board
Assembled & Tested
1 MHZ - No extended addressing
Can be set up for $0-7FFF or 8000-FFFF

$149.95

MasterCard VISA

DATA-COMP SOUTH EAST MEDIA

P.O. Box 794 *HIXSON, TN 37343*

1-615-842-4601

Verbatim Diskettes.

5" Soft Sector Disks

Single Side Single Density	$2.75 ea.
Single Side Double Density	$2.75 ea.
Double Side Double Density	$4.92 ea.
Plastic Storage Box	$2.00 ea.

8" Soft Sector Disks

Single Side Single Density	$3.75 ea.
Single Side Double Density	$4.10 ea.
Double Side Double Density	$4.75 ea.
Plastic Library Box	$5.00 ea.

Foreign Orders Add 10% Surface—20% Air Mail

DRIVES & CABINETS W/PS

5 1/4"

TANDON - Single Sided, Double Density, 40 Track.....$249.95

TANDON - Double Sided, Double Density, 40 Track.....$349.95

CABINET - Single Drive with Power Supply.....$ 79.95

CABINET - Double Drive with Power Supply.....$ 99.95

CABLE - Single Drive.....$ 24.95

CABLE - Double Drive.....$ 34.95

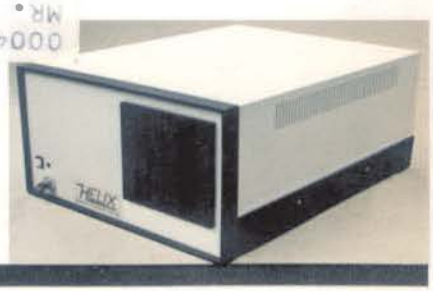
* NOTE - When ordering cables please specify S50 Bus or Other!!!

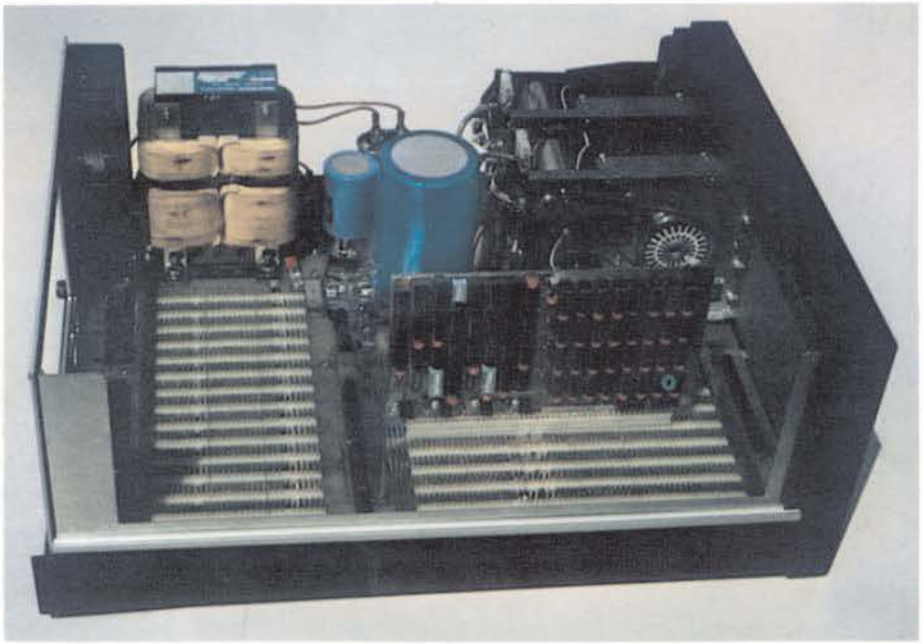
Call or write for disk controller Board information.

68 Micro Journal
5900 Cassandra Smith Rd.
Hixson, TN 37343

Second Class Postage Paid
At Chattanooga, TN
ISSN 0194-5025

HELIX™





THE MAINFRAME

- Industry Standard Optima™ Cabinet
- Largest Constant Voltage Power Supply in the industry
- S-64 Bus gives 16 Bit Power and S-50 Bus Compatibility
- 10 Main (S-64) Slots
- 14 I/O (S-30) Slots plus 2 On-board
- On-board Baud Rate Generator to 38.4Kb
- Space and Power for two 5¼" Disk Drives
- Full Address Decoding for I/O Slots
- Two RS-232 Serial and Two parallel Ports On-board
- Single Board Construction for Reliability
- Faraday Shielded Bus Lines give "Text Book Clean" Signals

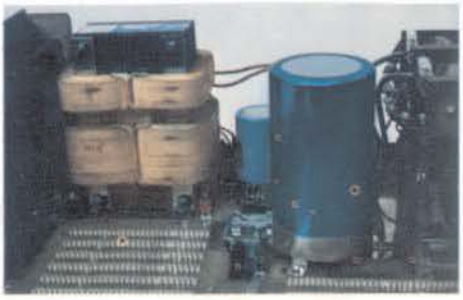
THE PROCESSORS

6809

- Standard 2 MHz Operation
- Standard DAT Compatible with GIMIX and SWTPC
- Standard 6840 Interval Timer
- Standard 1K Scratchpad RAM
- Standard Clock/Calendar with Battery
- Provision for Programmers Console

68000

- Standard 8 MHz Operation
- Memory Management Hardware
- Provision for Programmers Console
- 16 Bit Power and 8 Bit Compatibility



The HELIX™ computer system represents the latest advance in S-50 bus computer systems. Relying on the physical nature of S-50 bus connectors to guarantee compatibility, the HELIX adds 14 bus lines (becoming S-64) to allow a 68000 processor to operate with full 16 bit data transfer and 24 bit addressing, while at the same time providing full interchangability with existing S-50 components.

Offered with a selection of processors, memories, and peripheral controllers, a HELIX system can be configured for applications ranging from advanced hobbyist to multiterminal time-sharing.

Designed to offer the utmost in speed, reliability, and utility at a reasonable price, it represents a new standard of quality for those who require a professionally designed computer for professional use.

THE POWER SUPPLY

- Ferro-resonant Transformer for Line Noise and Under-Voltage Protection
- Conservative 25 Amps at 8.5 Volts
- Conservative 5 Amps at ±16 Volts
- Conservative Component Rating for Reliability

THE COMPONENTS

- Fully Socketed
- Gold Plated Bus Connectors
- Only "B" Series 68XX Components Used
- Only Top Grade Logic Circuits Used
- Industrial Grade Components Throughout

THE MEMORIES

DM-64

- Field Proven
- Proprietary Memory Control Logic
- Fully Transparent Refresh
- Tested at 2.5 MHz Operation

DM-512

- 512K Bytes on a Single S-64 Board
- 16 Bit Power and 8 Bit Compatibility
- Runs in Existing S-50 Systems where Physical Space Allows
- Full 24 Bit Addressing
- Fully Transparent Refresh

THE PRICES

Because of the variety of configurations possible, full pricing cannot be given. Representative prices are:

- 64K 6809 HELIX $1995
- 64K 68000 HELIX $2595
- 512K 6809 HELIX $4450
- 512K 68000 HELIX $4995

HAZELWOOD COMPUTER SYSTEMS

907 E. Terra, O'Fallon, Missouri 63366 (314) 281-1055

Dealer and OEM Inquiries Invited. We support our Dealers.

Optima is a Trademark of Scientific-Atlanta